Craft: Final Report

Martin Fagerhus (mf2967)
Roy Prigat (rp2719)
Daniel Tal (dt2479)
Abhijeet Mehrotra (am4586)

Contents

1 Introduction

Craft is a programing language geared towards creating a two dimensional grid-like game world. Its purpose is to allow users to easily define a complex game using basic syntax. The programmer is given the ability to define the size and color of a world which is the basic underlying structure of a game. He/she is also given the ability to easily add elements and events to create a dynamic and interactive environment.

This can be seen as a foundational layer for easy implementation of a functional two-dimensional grid-like game and simultaneously provides opportunity for more complex games with a multitude of features. Our language will simplify game creation by abstracting away the need for the programmer to account for threading and multiple other lower-level features that are imperative for creating games in other languages. The game developer can therefore spend more time on creative aspects of the game and less time on implementing tedious back-end functionality.

2 Tutorial

2.1 Setup

In order to setup the compiler, OCaml llvm library needs to be installed, ideally using the package manager OPAM. We also need to run the C interface: glib: 2.54.2, sdl: 1.2.15 and pkg-config to link glib. The steps are mentioned in the README.md file.

2.2 Using the complier

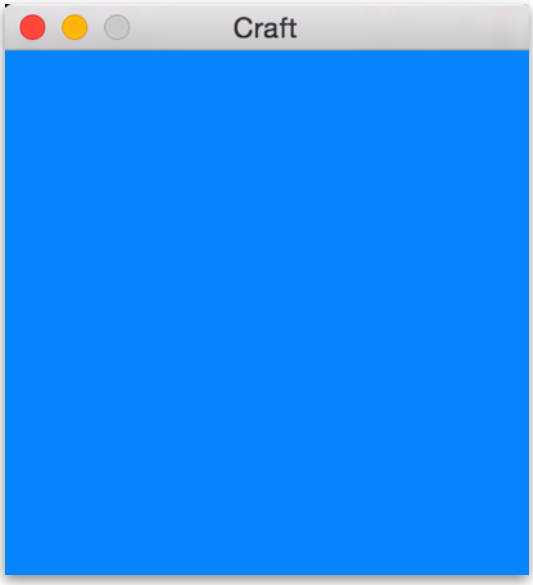
First, run **make** inside the source folder in order to generate the runtime environment and the Craft to LLVM IR compiler, **craft.native**, which takes in Craft source files and outputs executable programs:

> make
> ./craft.native <filename.crf>
> ./filename.exe

2.3 Helloworld

The most basic structure of a Craft program includes a definition of a world with its basic properties(size and color). Once this is created, the programmer can add things such as elements to create dynamic and static objects such as balls, enemies, walls, etc. These are all created on top of the world. Below is a basic hello world program that shows the creation of an unpopulated world with nothing but its properties (color and size).

```
world {
     properties {
        size = (250,250);
        color = "3399ff";
    }
}
```



2.4 World

The world essentially defines the window size and color of the game, on top of which all elements and event will be bound. This is a mandatory part to any Craft program.

2.4.1 Properties

Within world, the programmer must define its color and size within the properties scope. This must be explicitly created within the world scope. Without this, or with duplicates of either of the two properties, an error will stop your program from compiling. Below is a basic properties block that must be within the scope of a world.

2.4.1 Call Elements and Events

Below the properties block but within the world is where the programmer must call for their elements and events to be added to the world. These will be written in the form shown below.

```
element player = new player(20,20);

add_event(move_up);
```

2.5 Elements

Elements are how a programmer creates players, enemies, and anything his game includes which will be used in either the game interface or just for visual representation. As previewed above, the game maker must call the element within the world scope outside of its properties giving a position on where the entity should be placed in the world. While this is true, an element must be created before the world structure in your program. An element can be given any user-defined feature although it must contain both a color and a size. below is a section of code in which an element is created with a size, color, direction, and speed.

```
element player1 {
    size = (14,14);
    color = "00ff00";
    direction = 70;
```

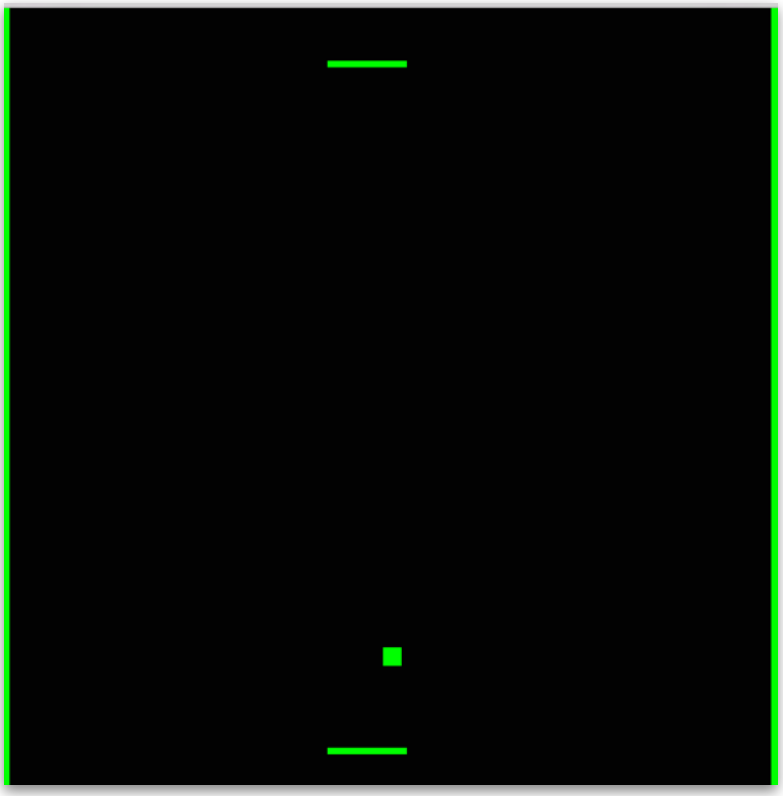
```
    speed = 2;
}
```

2.6 Events

The final Craft defined structure is the event block. This block contains the rules for the way an element can interact within the world. Within it is a condition related to its movement when a key is pressed as well as an action which defines what that condition does to the bounded element. Below is an example.

```
event move_right(paddle1) {
    condition = key_press("RIGHT");
    action {
      paddle1.pos.x = paddle1.pos.x + 5;
    }
}
```

That is all it takes! With these three simple structures, any programmer can create a game, from something as simple as a maze to two player pong or even space invaders!



3 Reference Manual

3.1 Lexical Conventions

3.1.1 Tokens

There are six classes of tokens: identifiers, keywords, constants, string literals, operators, and other separators. Spaces, tabs, and newlines can be used interchangeably to separate tokens. Some white space is required to separate otherwise adjacent identifiers, keywords, and constants.

3.1.2 Comments

Similar to the Python language, the character # introduces a single line comment which terminates with a new line character. As a consequence, the language does not support nested comments. Commented lines are ignored by the scanner and are not recognized as tokens.

3.1.3 Identifiers

Identifiers must begin with a lowercase alphabetic letter, followed by any sequence of alphanumeric characters. Identifiers can be of any length and letter case is significant.

3.1.3 Keywords

The following identifiers and symbols are reserved for use as keywords:

```
new    def    if    else   while   properties
element   world   color   pair   size   event
  int   bool  key_press   action   condition
          true   false   pos   !!
```

3.1.4 Literals

Literals are constant values of some built in primitive types which include: int, boolean and string.

3.1.4.1 Integer Literal

Constants that are a sequence of digits.

3.1.4.2 Boolean Literal

Represented by the keywords true and false.

3.1.4.3 String Literal

A string literal consists of a collection of characters enclosed in double quotes "…". It is not possible to index through a statically declared string nor is it possible to manipulate the data contained in the string.

3.2 Types

3.2.1 Primitives

3.2.1.1 Integers

Numbers of Integer type are declared with the int keyword.
Syntax: `int <id> = <int literal>;`
Example: `int x = 3;`

3.2.1.2 Booleans

Boolean values are declared with the bool keyword and are assigned either true or false values.
Syntax: `bool <id> = <boolean literal>;`
Example: `bool yes = true;`

3.2.2 Non-Primitives

3.2.2.1 Pair

A pair value is defined using two integer values, separated by a comma, and enclosed by parentheses. Anything except natural numbers (nonnegative) will be rejected as well as any pair values that exceed the game grid size.

Syntax: `pair <id> = (<int>,<int>);`
Example: `pair p = (3,3);`

3.2.2.2 Color

A color value is essentially a string literal enclosed by quotation marks. The string literal has to represent a hex value(semantically checked by the compiler).
Syntax: `color <id> = "<string>";`
Example: `color c = "ffffff";`

3.4 Expressions

3.4.1 Binary operators

Craft supports the following binary operations:

```
+    -    /    *   !=
&&   ||   <    >    ≤    ≥
```

3.4.2 Unary operators

Craft supports the following unary negation operations:

```
!   -
```

3.4.3 Assignment

Assignment is a right associative binary expression in, and also the operator with the lowest precedence. The right hand side is evaluated and stored in the location on the left hand side. This means that the left hand side must either be an identifier, a bracketed access, or a dot access expression.

3.4.4 Function calls

Like in most languages, when calling a function, the programmer must give the function name, followed by parentheses. Within the parentheses the correct number of arguments must be given for the function to run. Once this happens the function returns what it has evaluated to using the given arguments.

3.4.5 Properties access

The access to properties occurs using the member access operator (.). This allows the programmer to access the properties of an element in order to change or just use them for some type of operation he/she chooses.

3.5 Statements

3.5.1 Single expression

```
0;
x;
```

3.5.2 Return

Example of return statement in global function.

```
def int foo(int x) {
 return x;
}
```

3.5.3 Control Flow

Implementation of basic control flow.

3.5.3.1 If/Else

```
bool x = true;
if (x) {
  x = false;
} else {
  x = false;
}
```

3.5.3.2 While loops

```
int x = 2;
       while (x < 3) {
              x = x + 1;
       }
```

3.5.3.3 For loops

```
 int x = 3;
 int y = 4;
 for (x ; x<4; x = x + 1) {
      y = y + 1;
 }
```

3.5.4 New

The new state is used to add elements to the world by inserting it into the world body.

```
element player1 {
   size = (5,50);
   color = "4268f4";
}

element player2 {
   size = (30,30);
   color = "4268f4";
}

world {
   properties {
       size = (500,500);
       color = "f4f441";
   }

   element player1 = new player1(100,100);
   element player2 = new player2(300,300);
}
```

3.5.5 Event call

The event call is a statement used in order to add events to the game and bind them to a specific element. They require a movement and must be placed within the scope of world. Two examples below:

```
add_event(move_right);
add_event(move_left);
```

3.6 Functions

3.6.1 Built in functions

3.6.1.1 add_event

This pre-defined function adds the event passed into the parameter to the global event loop that runs in the global loop at every clock tick.

3.6.2 User defined functions

Syntax	Example

`def <typ> <id>(<typ><id>…) {`
`  <stmt list>`
`}`
	`def int test(int a) {`
	`  return a;`
	`}`

3.7 Events

Events help define the rules by which the game will abide.

Syntax	Example

`event <id>(<id>,…) {`
` condition = <expr>;`
` action {`
`  <stmt>`
` }`
`}`
	`event move_up(player) {`
	` condition = key_press("UP");`
	` action {`
	`  player.pos.x = player.pos.x + 100;`
	` }`
	`}`

3.8 Elements

Elements are the building blocks of the game

Syntax	Example

element <id> {
  size = (int,int);
  color = "string";
}
	element <id> {
	  size = (5,5);
	  color = "aaaaaa";
	}

3.9 World

The world and its basic properties is the only required component of a Craft program and exactly one may be defined.

Syntax	Example

world {
  properties {
  size = (int,int);
  color = "string";
  }
}
	world {
	  properties {
	  size = (100,100);
	  color = "ffffff";
	  }
	}

3.10 Program Structure

The following represents the overall layout of a proper program using craft:

```
<global variables> - optional
<global functions> - optional
```

```
<event definitions> - optional
    <Event>
        <condition>
        <action>

<element definitions> - optional
    <element>
        <properties>

<world definition> - required
    <world>
        <properties>
        <local variables> - optional
        <statements block> - optional
```

As can be seen, the only required aspect of a craft program is the definition of a world and its properties. Without it, the program will not run.

3.11 Grammar

program:
var_decl_list func_decl_list event_list element_list world

var_decl_list:
ε | var_decl_list var_decl

var_decl:
typ ID ASSIGN expr SEMI

typ:
INT | FLOAT | BOOL | COLOR | PAIR

formals_list_opt:
ε | formals_list

formals_list:
typ ID | formals_list COMMA typ ID

event_formals_list:
ID ID | event_formals_list COMMA ID ID

actuals_opt:
ε | actuals_list

actuals_list:

expr | actuals_list COMMA expr

func_decl_list:
ε | func_decl_list func_decl

func_decl:
DEF typ ID LPAREN formals_list_opt RPAREN LBRACE var_decl_list stmt_list RBRACE

property_list:
ε | property_list property

property:
var_decl | SIZE ASSIGN expr SEMI | COLOR ASSIGN expr SEMI | DIRECTION ASSIGN expr SEMI | SPEED ASSIGN expr SEMI

event_list:
ε | event_list event

event:
EVENT ID LPAREN event_formals_list RPAREN LBRACE COND ASSIGN expr SEMI ACT LBRACE stmt_list RBRACE RBRACE

element_list:
ε | element_list element

element:
ELEMENT ID LBRACE property_list RBRACE

world:
WORLD LBRACE PROPS LBRACE property_list RBRACE var_decl_list stmt_list RBRACE

stmt_list:
ε | stmt_list stmt

stmt:
expr SEMI | element_decl | RETURN expr SEMI |
LBRACE stmt_list RBRACE | IF LPAREN expr RPAREN stmt %prec NOELSE
| IF LPAREN expr RPAREN stmt ELSE stmt
| WHILE LPAREN expr RPAREN stmt | FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt | ID LPAREN ID RPAREN SEMI

element_decl:
ELEMENT ID ASSIGN NEW ID expr SEMI

expr:
literals | expr PLUS expr | expr MINUS expr | expr TIMES expr | expr DIVIDE expr | expr EQ expr | expr NEQ expr | expr LT expr | expr LEQ expr | expr GT expr | expr GEQ expr | expr AND expr | expr OR expr | NOT expr | MINUS expo %prec NEG | expr ASSIGN expr
| ID PERIOD POS PERIOD expr | ID LPAREN actuals_opt RPAREN |
| ID LPAREN ID LPAREN actuals_opt RPAREN RPAREN |
| LPAREN expr RPAREN | LPAREN expr COMMA expr RPAREN |

```
    | KEY_PRS LPAREN expr RPAREN

literals:
    INT_LITERAL | FLOAT_LITERAL | STRING_LITERAL | TRUE | FALSE |ID |
    COLOR | SIZE

expr_opt:
    ε | expr
```

4 Project Plan

4.1 Team Members

At the beginning of the semester, project responsibilities were assigned to each of the four members. And yet, throughout the semester, depending on the stage of the project each member has contributed to multiple parts of the project.

Daniel: Testing, Semantics

Roy: AST, Frontend

Abhijeet: C interface, Codegen callbacks, C Testing

Martin: Codegen, Testing

Everyone helped by pair programming and jumped in to debug and brainstorm when needed.

4.2 Process

4.2.1 Planning

Regular checkins and meetings were held through the course of the semester. They picked up pace in focus and productivity as we got a better idea about the project as the semester progressed from the class and our own research.
We had to remedy and change stuff when our action block problem as discussed above got us stuck near the end of the semester when everyone was really busy. But, we managed to rewrite stuff and work around it to get a functional project.

4.2.3 Development

In the early stages, we focussed a lot on getting the design of the language right and getting a functional scanner, parser and AST. We built out the C interface and tested it separately. And then we focussed on getting codegen right. Things evolved as we modified features and implementations.

4.2.3 Testing

Due to the graphical nature of the language, testing was really hard till the while thing came together and was mostly manual. We did test the c interface separately using a driver function and stress tested the semantics.

4.3 Timeline

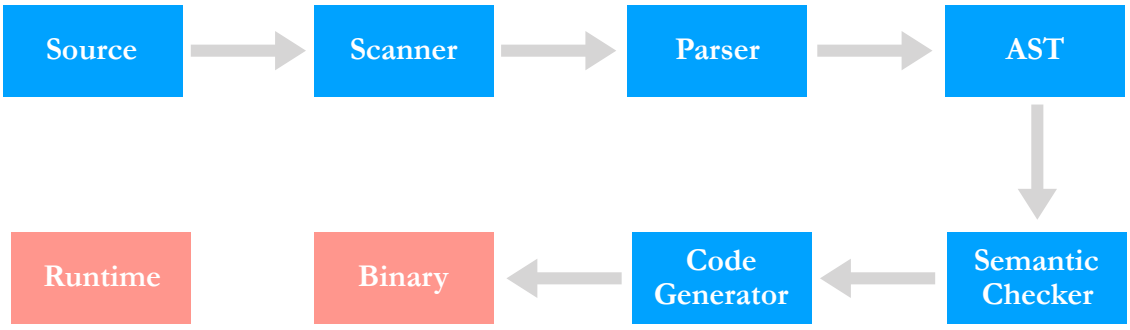
The following timeline represents the actual milestones reached throughout the semester:

Week	Milestone	Due
Sep 25 - Sep 29	Brainstorm language theme	Proposal
Oct 2 - Oct 6	Define expected features and syntax	
Oct 9 - Oct 13		
Oct 16 - Oct 20		LRM
Oct 23 - Oct 27	Build testing suite skeleton	
Oct 30 - Nov 3	Scanner and Parser started	
Nov 6 - Nov 10	Exploring how to use SDL and first setup of runtime environment	Helloworld
Nov 13 - Nov 17	Scanner and parser near completion, begin implementation of AST	
Nov 20 - Nov 24	Helloworld program running	
Nov 27 - Dec 1	Implement global vars	
Dec 4 - Dec 8	Finalize language features, implement world and elements	

Dec 11 - Dec 15	Implement events and keypress, added basic testing files	
Dec 18 - Dec 20	Testing suite automation, final report assembled, code generation and semantics complete	Presentation

5 Architectural Design

5.1 Block Diagram



5.1 Scanner

Filename: scanner.mll

Written in OCamllex, the scanner iterates through the Craft source file (.crf extension) and tokenizes it into keywords, identifiers and literals, disregarding commented lines of code. It does not scan any syntactically invalid, but rather throws an error for such a case. The tokens created by the scanner are then used by the parser to create the abstract syntax tree.

5.2 Parser and AST

Filenames: parser.mly, ast.ml

Written in OCamlyacc, the parser receives the stream of tokens produced by the scanner and generates an abstract syntax tree(AST). If the tokens are successfully parsed, then the code is syntactically correct. The ast.ml file defines the components of the language by which the parser is guided in order to create a proper syntax tree.

5.4 Semantics

Filename: semant.ml

Written in OCaml, the semantics checker iterates over the code stored in the abstract syntax tree and assures it is semantically correct — i.e that the program is meaningful and properly defined on all its components(expressions, statements, functions and all variations of operations).

5.5 Codegen

Filename: codegen.ml

Written in OCaml, the code generator takes the semantically checked AST and generates the corresponding LLVM IR.

5.6 Runtime

The C interface provides the run loop and graphics interface for the language. It exposes an interface which codegen can call. It uses SDL for the graphics rendering and Glib for the library functions.

It is able to keep track of all the elements, the world and the callback functions. It updates the values of elements as required and triggers the callback functions to trigger actions bound to them.

It also takes care of rendering defined speeds and directions for moving objects. All collisions for moving objects are perfectly elastic in nature. Motion imparted using key bindings is non elastic in nature.

It also provides a function to codegen to be able to detect collisions.

6 Test Plan

6.1 Automated tests

After making the compiler, in order to automatically run tests on things such as expressions, semantics, declarations, statements, and functions you must run the testall.sh shell script as follows:

```
> ./testall.sh
```

This automatically runs all the tests and outputs a log called testall.log. In it, if a test passed you will get a #SUCCESS. If it failed, the programmer will be able to see the exceptions or reasons for failure for each file under the file name.

6.2 Manual tests

Due to the heavy reliance on a graphical interface in Craft, the programmer must perform many manual tests in order to ensure that their program outputs the exact details they expect of their world. As an example, one way this can be done includes, creating an element within a loop or conditional in order to make sure the program passes through it.

7 Lessons Learned

7.1 Reflection

We took on the challenge on making a 2D gaming language since everyone seemed genuinely excited about the idea. The graphics interface provided an unique set of challenges which forced us to scope down the project for practical considerations. This largely due to us being unable to implement control flow blocks in LLVM from OCaml efficiently. This to be a blocking aspect for all of our project development.

We are able to create action blocks in LLVM after parsing them from the AST and trigger them individually from the C-interface but we were unable to access

dynamically allocated memory for a non-specified number of events. This forced us to offload the functionality of dynamically generated action blocks onto a set of pre-specified set of actions in the C-interface. We are able to successfully fill them up with variables specified in the language and trigger them from the C-interface. We are able to successfully support configurable speed, direction, and elastic collisions.

in the future we would love to complete the true functionality that we had decided to build, that is, being able to overload collisions and have arbitrary action blocks. We were able to pass configured structs for elements, world and events and get the callback functions working but ran into issues when we wanted to dynamically manipulate elements in the world through the code generation stage.

```
struct element{
    char * name;
    struct tuple size;
    struct tuple position;
    char* el_color;
    // void (*event_fn)(struct element*);
    int direction;
    int speed;
};
```

The event function was supposed to be the pointer to the action block for each event-element binding but we had to scope that down.

```
    | A.ECall ("add_event", event_name) ->

        let event = StringMap.find (event_name ^ "_event") events_helper_map
in
        let condition = string_of_expr event.A.condition in (*ex: "UP"*)

        let first_elem = List.hd event.A.eformals in (*it's a string. ex
player which is now the name*)

        let event_func_type = L.function_type (L.void_type context) [||] in
        let event_func = L.define_function (condition ^ "_event_func")
event_func_type the_module in
        let event_builder = L.builder_at_end context (L.entry_block
event_func) in (*event_func runs a new basic block. trigggered by fucntion
pointer in C*)

        let cond_str_ptr = L.build_global_stringptr condition
("condition_str_ptr") event_builder in
        let elem_name_str_ptr = L.build_global_stringptr first_elem
(first_elem ^ "_str_ptr") event_builder in

        (*call the c function and get a yes or no*)
```

```
        let return_val = L.build_call is_key_pressed_func [|cond_str_ptr|] ""
event_builder in (*stored in new basic block*)

        let is_pressed_bb = L.append_block context ("pressed_" ^ condition)
event_func in
        let merge_bb = L.append_block context ("merge_" ^ condition)
event_func in

        let is_pressed_builder = L.builder_at_end context is_pressed_bb in

        ignore(L.build_call move_func [|elem_name_str_ptr;cond_str_ptr|] ""
is_pressed_builder); (*tell C to move element*)
        ignore (L.build_br merge_bb is_pressed_builder);

        let compare_instruction = L.build_icmp L.Icmp.Eq return_val
(L.const_int i32_t 1) ("key_pressed_"^condition) event_builder in
        ignore(L.build_cond_br compare_instruction is_pressed_bb merge_bb
event_builder);

        let merge_builder = L.builder_at_end context merge_bb in
        ignore (L.build_ret_void merge_builder);

        ignore (L.build_call add_event_func [|event_func|] "" builder);
(*giving C the pointer and it will call the func pointer*)

        builder
```

We specifically wanted to run the stmt function on the action statements in the event inside event_builder which would have be able to retrieve and update element values during runtime since we had added functionality for it in the expr function but we could not get the event_builder to access the correct memory hence this functionality was lost. We tested the functionality by hard coding in element manipulation in the correct builder and made it work but unfortunately we could not extend this to be triggered by the callback functions from the C-interface. Hence some of our element manipulation is unused from the code generation stage, example of this below.

```
 | A.PAccess (s1,s2,s3) ->
      (match s3 with
        | "x" ->
          (match s2 with
            | "pos" -> let elem_ptr = StringMap.find (s1 ^ "_element") map in
                       let pos_ptr = L.build_struct_gep elem_ptr 2
("elem1_pos_ptr") builder in
                         let x_ptr = L.build_struct_gep pos_ptr 0 ("x_ptr")
builder in
                           L.build_load x_ptr "x" builder (*return the x
value in llvm type*)

            | "size" -> let elem_ptr = StringMap.find (s1 ^ "_element") map
in
                         let size_ptr = L.build_struct_gep elem_ptr 2
("elem2_size_ptr") builder in
                           let x_ptr = L.build_struct_gep size_ptr 0
("x_ptr") builder in
```

```
                              L.build_load x_ptr "x" builder
            | _ -> L.const_int i32_t 0
          )
        | "y" ->
          (match s2 with
            | "pos" -> let elem_ptr = StringMap.find (s1 ^ "_element") map in
                       let pos_ptr = L.build_struct_gep elem_ptr 2
("elem3_pos_ptr") builder in
                         let y_ptr = L.build_struct_gep pos_ptr 1 ("y_ptr")
builder in
                           L.build_load y_ptr "y" builder (*return the y
value in llvm type*)
            | "size" -> let elem_ptr = StringMap.find (s1 ^ "_element") map
in
                         let size_ptr = L.build_struct_gep elem_ptr 2
("elem4_size_ptr") builder in
                           let y_ptr = L.build_struct_gep size_ptr 1
("y_ptr") builder in
                             L.build_load y_ptr "y" builder
            | _ -> L.const_int i32_t 0
          )
        | _ -> L.const_int i32_t 0
```

7.2 Martin

It has been a big challenge to implement our language due to many factors. We decided early on that we wanted implement a 2D game language that would make game design simple and enjoyable for the programmer. In the beginning it was challenging to understand exactly what we had to do in order to implement it and a lot of the early meetings were pressed by this issue. I personally felt that this lead to a little progress on the programming aspect in the beginning of the semester and much of the time was spent of discussing syntax of the language and high level ideas. It would have been helpful to work through all aspects of the project together on earlier stages instead of dividing work since the knowledge about various parts was a bit spread out and it would have been possible to work more efficiently if everyone had were on the same page earlier on. With that said its been a very good learning experience.

Advice: Starting early was obviously an advice we should have taken more seriously. I general I think that the best way for a group to succeed in this project is to start looking at the code very early on and try to implement get everyone to work together on implementing "Hello World" as early as possible. We thought it would be easier to meet and discuss high level strategy and then work separately to implement specific features but I think it would have been easier if everyone walked went over early versions of Craft together which would have allowed for easier developing later on. It is really important to start working with the code early in order to become familiar

with OCaml and LLVM IR. Getting the basic understanding of all aspects of the compiler as early as possible will of great help.

7.3 Roy

The task of designing and implementing a programming language is challenging given any circumstance, particularly under a very limited amount of time. Even agreeing early on basic syntax, features and a general language theme could be difficult without prior experience. While we were attempting to be efficient in assigning specific roles to each team member, I realized that working only on certain parts of the compiler kept me out of sync from other parts implemented by my teammates. I learned that problems and issues in the compiler propagate through its different layers, and that it would have been easier to solve them if each member had been more involved in all aspects of the compiler's implementation. More importantly, it would have allowed the team to be more flexible and continue full feature implementations when a member wasn't available.

Advice: being ambitious in choosing your language features and theme may be very tempting at first, but when Ocaml and LLVM are new to all team members it may be wise to aim for a "simple" language that has the potential to expand and include some exclusive features. Thoroughly read MicroC and fully understand its implementation, then make sure your language has the same basic features implemented before you proceed to add more fancy and complex features. Finally, let each team member add new features to the language from the beginning of the complier chain to the end. Not only will this approach allow the team to keep in synch on all levels of the complier, but it will also assure that if a team member isn't available, others can substitute and continue his work.

7.4 Daniel

Learning a language (possibly two), building a language/compiler in that language, and working on it simultaneously in a team is no trivial task! Throughout the process, I think the most important thing learned is start small. Build your compiler iteratively across the board and not by file. While doing this, make sure all members of the group are caught up and understand the nuances of each file as well as the way the files interact to produce a compiler that can understand your language. In this sense,

anything that needs to be changed throughout the program can not only be done by any group member but also understood by all.

Advice: Although this is a daunting task, the most important thing to do is start early. Get on top of the OCaml syntax, and make sure to understand the ins and outs of functional programming. Once this is done look to examples such as MicroC to better understand the way different files merge and interact within your compiler. In regards to team, it is important to be very transparent with team members and have good communication across the board. Be honest with each other and open to each individual's ideas and advice. This way everyone can get along and be productive simultaneously. Lastly, before you even start to code, try to visualize your project and try to agree on your language syntax, semantics, and what you want it to do. This sort of planning will save you a lot of time in the future when you try to implement added features.

7.5 Abhijeet

Building a language is always a hard task. Or initial setback was wrapping our heads around Ocaml. Once that got sorted, we had very little idea on how to get codegen and the c interface talking to each other.
We took sometime to come up with the idea of using callback functions to trigger functionality and responses. The hardest part where we has to cut back on features was making the basic block decision trees to interface with the c engine. We faced lots of issues generating them from Ocaml and the non abundance of such documentation just made our task harder. We failed to catch this part from the beginning and research it more. This would have been avoided by more end to end and non hardcoded testing of features. I was involved in debugging this in the end and it required a significant rewrite of codegen.
Being involved in making features and debugging and figuring out design and other modules gave me a greater appreciation of the overall project.

Advice: Initially the language was an intimidating task and it took us quite sometime to get an idea of how to actually start wiring stuff together with codegen. We should have focussed more on end to end testing too. Mocking separate parts for each module underestimated the difficulty of getting stuff running with codegen and we had to scale back on features to balance it out. Also, all members have to be involved from the beginning in codegen. Helps avoid wasting time on getting everyone upto speed in the end to solve a blocker issue.

8 Appendix

8.1 Compiler

ast.ml

```
(* CRAFT Abstract Syntax Tree *)

module StringMap = Map.Make(String)

type op = Add | Sub | Mult | Div | Equal | Neq | Less | Leq | Greater
| Geq | And | Or
type uop = Neg | Not
type typ = Int | Float | Bool | Void | Pair | Color | String
type bind = typ * string

type expr =
    ILiteral of int
  | FLiteral of float
  | SLiteral of string
  | BLiteral of bool
  | Pr of expr * expr
  | Cr of expr
  | Id of string
  | Binop of expr * op * expr
  | Unop of uop * expr
  | Assign of expr * expr
  | PAccess of string * string * string
  | CAccess of string * string
  | Keypress of expr
  | Call of string * expr list
  | Noexpr

type element_decl = string * string * expr

type stmt =
    Block of stmt list
  | Expr of expr
  | Return of expr
  | If of expr * stmt * stmt
  | Condition of stmt * stmt
  | While of expr * stmt
  | For of expr * expr * expr * stmt
  | New of element_decl
  | ECall of string * string

(* Variable declaration  *)
type var_decl =  typ * string * expr

(* Event formals *)
```

```
type event_formal = string

(* Function declaration *)
type func_decl = {
  typ : typ;
      fname : string;
      formals : bind list;
      locals : var_decl list;
      body : stmt list;
}

(* Events *)
type event = {
      evname : string;
  eformals : event_formal list;
  condition : expr;
  action : stmt list;
}

(* Elements *)
type element = {
      ename: string;
      e_properties: var_decl list;
}

(* World *)
type world = {
  w_properties: var_decl list;
  init_locals : var_decl list;
  init_body: stmt list;
}

(* Program *)
type program = var_decl list * func_decl list * event list * element
list * world



(* AST Print functions *)

let string_of_typ = function
  Int -> "int"
| Float -> "float"
| String -> "string"
| Bool -> "bool"
| Void -> "void"
| Pair -> "pair"
| Color -> "color"

let string_of_op = function
  Add -> "+"
| Sub -> "-"
| Mult -> "*"
| Div -> "/"
```

```
| Equal -> "=="
| Neq -> "!="
| Less -> "<"
| Leq -> "<="
| Greater -> ">"
| Geq -> ">="
| And -> "&&"
| Or -> "||"

let string_of_uop = function
  Neg -> "-"
| Not -> "!"

let rec string_of_expr = function
  ILiteral(l) -> string_of_int l
| FLiteral(l) -> string_of_float l
| BLiteral(true) -> "true"
| BLiteral(false) -> "false"
| SLiteral(s) -> s
| Pr(x,y) -> "(" ^ string_of_expr x ^ "," ^ string_of_expr y ^ ")"
| Cr(c) -> string_of_expr c
| Id(s) -> s
| Binop(e1, o, e2) ->
  string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2
| Unop(o, e) -> string_of_uop o ^ string_of_expr e
| Assign(v, e) -> string_of_expr v ^ " = " ^ string_of_expr e
| Call(f, el) ->
  f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"
| PAccess(s1,s2,e) -> s1 ^ "." ^ s2 ^ "." ^ e
| CAccess(s1,s2) -> s1 ^ "." ^ s2
| Keypress(e) -> "key_press(" ^ string_of_expr e ^ ")"
| Noexpr -> ""

let rec string_of_stmt = function
    Block(stmts) ->
      "{\n" ^ String.concat "" (List.map string_of_stmt stmts) ^ "}\n"
  | Expr(expr) -> string_of_expr expr ^ ";\n";
  | Return(expr) -> "return " ^ string_of_expr expr ^ ";\n";
  | If(e, s, Block([])) -> "if (" ^ string_of_expr e ^ ")\n" ^
string_of_stmt s
  | If(e, s1, s2) ->  "if (" ^ string_of_expr e ^ ")\n" ^
      string_of_stmt s1 ^ "else\n" ^ string_of_stmt s2
  | While(e, s) -> "while (" ^ string_of_expr e ^ ") " ^
string_of_stmt s
  | For(e1, e2, e3, s) ->
      "for (" ^ string_of_expr e1  ^ " ; " ^ string_of_expr e2 ^ " ; "
^
      string_of_expr e3  ^ ") " ^ string_of_stmt s
  | New(a,b,c) -> "element " ^ a ^ " = new " ^ b ^ string_of_expr c ^
";\n"
  | Condition(st1, st2) -> string_of_stmt st1 ^ string_of_stmt st2
  | ECall(f,evnt) -> f ^ "(" ^ evnt ^ ")"

let string_of_vars = function
```

```
  (t,s,e) -> string_of_typ t ^ " " ^ s ^ " = " ^ string_of_expr e ^ ";
\n"

let string_of_args (t, id) = string_of_typ t ^ " " ^ id ^ ";\n"

let string_of_event_formals (id) = id

let string_of_fdecl fdecl =
  "\ndef " ^ string_of_typ fdecl.typ ^ " " ^
  fdecl.fname ^ "(" ^ String.concat ", " (List.map snd fdecl.formals)
^
  ") {\n" ^
  String.concat "" (List.map string_of_vars fdecl.locals) ^
  String.concat "" (List.map string_of_stmt fdecl.body) ^
  "}\n"

let string_of_events event =
  "\nevent " ^ event.evname ^ "(" ^ String.concat ""
  (List.map string_of_event_formals event.eformals) ^ ") " ^
  "{\n " ^ "condition = " ^ (string_of_expr event.condition) ^ ";\n" ^
  "action {\n" ^ String.concat "" (List.map string_of_stmt
event.action) ^
  "\n}\n"

let string_of_elems elem =
  "\nelement " ^ elem.ename ^ " " ^
  "{\n " ^
  String.concat " " (List.map string_of_vars elem.e_properties) ^
  "}\n"

let string_of_world world =
  "\nworld {\nproperties {\n" ^
  String.concat "" (List.map string_of_vars world.w_properties) ^ "}
\n" ^
  String.concat "" (List.map string_of_vars world.init_locals) ^
  String.concat "" (List.map string_of_stmt world.init_body) ^
  "}\n"

let string_of_program (globals, funcs, events, elems, world) =
  String.concat " " (List.map string_of_vars globals) ^
  String.concat " " (List.map string_of_fdecl funcs) ^
  String.concat " " (List.map string_of_events events) ^
  String.concat " " (List.map string_of_elems elems) ^
  string_of_world world
```

scanner.mll

```
(* CRAFT Scanner *)

{ open Parser }

rule token = parse
  [' ' '\t' '\r' '\n'] { token lexbuf }     (* Whitespace *)
  | '#'                { comment lexbuf }   (* Comments *)
  | '('          { LPAREN }
  | ')'          { RPAREN }
  | '{'          { LBRACE }
  | '}'          { RBRACE }
  | ';'          { SEMI }
  | ':'          { COLON }
  | ','          { COMMA }

  (* Operators *)
  | '+'          { PLUS }
  | '-'          { MINUS }
  | '*'          { TIMES }
  | '/'          { DIVIDE }

  (* More Operators *)
  | '='          { ASSIGN }
  | "=="         { EQ }
  | "!="         { NEQ }
  | '<'          { LT }
  | "<="         { LEQ }
  | ">"          { GT }
  | ">="         { GEQ }
  | "&&"         { AND }
  | "||"         { OR }
  | "!"          { NOT }

  (* Special Operators *)
  | "."            { PERIOD }
  | "!!"         { COLLIDE }

  (* Control Flow *)
  | "if"         { IF }
  | "else"       { ELSE }
  | "while"      { WHILE }
  | "for"        { FOR }

  (* Keywords *)
  | "int"        { INT }
  | "float"      { FLOAT }
  | "bool"       { BOOL }
  | "void"       { VOID }
  | "true"       { TRUE }
  | "false"      { FALSE }
  | "size"       { SIZE }
  | "color"      { COLOR }
```

```
  | "speed"     { SPEED }
  | "direction" { DIRECTION }
  | "pos"        { POS }
  | "new"       { NEW }
  | "action"    { ACT }
  | "condition" { COND }
  | "return"    { RETURN }

  | "properties" { PROPS }
  | "event"      { EVENT }
  | "element"    { ELEMENT }
  | "world"      { WORLD}
  | "def"        { DEF }

  (* I/O Keywords *)
  | "key_press" { KEY_PRS }


  | ['0'-'9']+ as lxm
{ INT_LITERAL(int_of_string lxm) }
  | ['0'-'9']*'.'['0'-'9']+ | ['0'-'9']+'.'['0'-'9']* as lxm
{ FLOAT_LITERAL(float_of_string lxm)}
  | '"' (['a'-'z' 'A'-'Z' '0'-'9'])* as lxm)
'"'            { STRING_LITERAL(lxm) }
  | ['a'-'z' 'A'-'Z']['a'-'z' 'A'-'Z' '0'-'9' '_']* as lxm   { ID(lxm)
}
  | eof                                                      { EOF }
  | _ as char { raise (Failure("illegal character " ^ Char.escaped
char)) }


  and comment = parse
    '\n' { token lexbuf }
    | _    { comment lexbuf }
```

parser.mly

```
/* Ocamlyacc parser for CRAFT */

%{ open Ast %}

%token LPAREN RPAREN LBRACE RBRACE SEMI COLON COMMA
%token PLUS MINUS TIMES DIVIDE
%token ASSIGN EQ NEQ LT LEQ GT GEQ AND OR NOT
%token PERIOD COLLIDE
%token IF ELSE WHILE FOR RETURN
%token INT FLOAT STRING BOOL VOID TRUE FALSE
%token SIZE DIRECTION COLOR PAIR SPEED POS NEW ACT COND
%token EVENT DEF PROPS ELEMENT WORLD START
%token KEY_PRS

%token <int> INT_LITERAL
%token <float> FLOAT_LITERAL
```

```
%token <string> STRING_LITERAL
%token <string> ID
%token EOF

%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN
%left OR
%left AND
%left EQ NEQ
%left LT GT LEQ GEQ
%left COLLIDE
%left PLUS MINUS
%left TIMES DIVIDE
%right NOT NEG
%left ACCESS

%start program
%type <Ast.program> program

%%

/* Entry point */
program:
  var_decl_list func_decl_list event_list element_list world EOF
  { (List.rev $1, List.rev $2, List.rev $3, List.rev $4, $5) }

/* Primitive types */
typ:
    INT   { Int }
  | FLOAT { Float }
  | STRING { String }
  | BOOL  { Bool }
  | VOID  { Void }
  | COLOR { Color }
  | PAIR  { Pair }

/* Variables*/
var_decl_list:
  { [] }
  | var_decl_list var_decl       { $2 :: $1 }

var_decl:
  typ ID ASSIGN expr SEMI    { ($1, $2, $4) }

/* Element declaration */
element_decl:
  ELEMENT ID ASSIGN NEW ID expr SEMI  { New($2,$5,$6) }

/* Functions */
func_decl_list:
  { [] }
  | func_decl_list func_decl     { $2 :: $1 }
```

```
func_decl:
  DEF typ ID LPAREN formals_list_opt RPAREN LBRACE var_decl_list
stmt_list RBRACE
  {{
    typ = $2 ;
    fname = $3;
           formals = $5;
           locals = List.rev $8;
    body = List.rev $9;
  }}

/* Function arguments */
formals_list_opt:
  { [] }
  | formals_list               { List.rev $1 }

formals_list:
    typ ID                     { [($1,$2)] }
  | formals_list COMMA typ ID  { ($3,$4) :: $1 }

actuals_opt:
    /* nothing */ { [] }
  | actuals_list  { List.rev $1 }

actuals_list:
    expr                    { [$1] }
  | actuals_list COMMA expr { $3 :: $1 }

/* Properties */
property_list:
  { [] }
  | property_list property        { $2 :: $1 }

property:
    var_decl                    { $1 }
      | SIZE ASSIGN expr SEMI        { (Pair, "size", $3) }
      | COLOR ASSIGN expr SEMI       { (Color, "color", $3) }
  | DIRECTION ASSIGN expr SEMI  { (Int, "direction", $3) }
  | SPEED ASSIGN expr SEMI      { (Int, "speed", $3) }

/* Event arguments */
event_formals_list:
    ID                          { [($1)] }
  | event_formals_list COMMA ID  { ($3) :: $1 }

/* Events */
event_list:
  { [] }
  | event_list event  { $2 :: $1 }

event:
  EVENT ID LPAREN event_formals_list RPAREN LBRACE COND ASSIGN expr
SEMI ACT LBRACE stmt_list RBRACE RBRACE
  {{
```

```
    evname = $2;
    eformals = $4;
    condition = $9;
    action = List.rev $13;
  }}

/* Elements */
element_list:
  { [] }
  | element_list element   { $2 :: $1 }

element:
     ELEMENT ID LBRACE property_list RBRACE
  {{
    ename = $2;
    e_properties = List.rev $4;
  }}

/* World */
world:
     WORLD LBRACE PROPS LBRACE property_list RBRACE var_decl_list
stmt_list RBRACE
     {{
    w_properties = List.rev $5;
    init_locals = List.rev $7;
    init_body = List.rev $8;
     }}

/* Statements */
stmt_list:          { [] }
  | stmt_list stmt    { $2 :: $1 }

stmt:
       expr SEMI
{ Expr $1 }
  | element_decl                                          { $1 }
     | RETURN expr SEMI
{ Return $2 }
     | LBRACE stmt_list RBRACE
{ Block(List.rev $2) }
     | IF LPAREN expr RPAREN stmt  %prec NOELSE
{ If($3, $5, Block([])) }
     | IF LPAREN expr RPAREN stmt ELSE stmt
{ If($3, $5, $7) }
     | WHILE LPAREN expr RPAREN stmt                                  {
While($3, $5)}
  | FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt { For($3,
$5, $7, $9) }
  | ID LPAREN ID RPAREN SEMI
{ ECall($1, $3) }

expr_opt:
  { Noexpr }
  | expr          { $1 }
```

```
/* Expressions */
expr:
    literals                                    { $1 }
      | expr PLUS expr                            { Binop($1, Add,
$3) }
  | expr MINUS expr                             { Binop($1, Sub, $3)
}
  | expr TIMES expr
{ Binop($1, Mult, $3) }
  | expr DIVIDE expr
{ Binop($1, Div, $3) }
  | expr EQ expr
{ Binop($1, Equal, $3) }
  | expr NEQ expr
{ Binop($1, Neq, $3) }
  | expr LT expr
{ Binop($1, Less, $3) }
  | expr LEQ expr
{ Binop($1, Leq, $3) }
  | expr GT expr
{ Binop($1, Greater, $3) }
  | expr GEQ expr
{ Binop($1, Geq, $3) }
  | expr AND expr
{ Binop($1, And, $3) }
  | expr OR expr
{ Binop($1, Or, $3) }
  | NOT expr
{ Unop(Not, $2) }
  | MINUS expr %prec NEG                            { Unop(Neg,
$2) }
  | expr ASSIGN expr                            { Assign($1, $3) }
  | ID PERIOD POS PERIOD ID                     { PAccess($1,"pos",
$5) }
  | ID PERIOD COLOR
{ CAccess($1,"color") }
  | ID PERIOD SIZE PERIOD ID                    { PAccess($1,"size",
$5) }
  | ID LPAREN actuals_opt RPAREN                { Call($1, $3) }
  | LPAREN expr RPAREN                                  { $2 }
  | LPAREN expr COMMA expr RPAREN               { Pr($2,$4) }
  | KEY_PRS LPAREN expr RPAREN                  { Keypress($3) }

literals:
       INT_LITERAL                                { ILiteral($1) }
     | FLOAT_LITERAL                          { FLiteral($1) }
  | STRING_LITERAL                { SLiteral($1) }
     | TRUE
{ BLiteral(true) }
     | FALSE
{ BLiteral(false) }
  | ID                                    { Id($1) }
  | COLOR                          { Id("color") }
```

```
  | SIZE                          { Id("size") }
```

exceptions.ml

```
exception MissingEOF
exception Invalid
exception InvalidBinaryOperation
exception InvalidAssignment
exception WrongNumberOfArguments
exception IncorrectArgumentType of string
exception DuplicateVariable of string
exception UndefinedId of string
exception UndeclaredId of string
exception IllegalBinOp of string
exception IllegalUnOp of string * string * string
exception IllegalPairType of string
exception PassedReturn of string
exception IncorrectType of string
exception UnrecognizedFunction of string
exception IncorrectColorType of string
exception IncorrectPairType of string
exception NonBooleanType
exception NonAccessibleProp of string
exception IllegalArgument
exception WrongKeyPress
exception IllegalCondition
```

semant.ml

```
(* Semantic checking for the CRAFT compiler *)

open Ast

module E = Exceptions
module StringMap = Map.Make(String)

let check (globals, funcs, _, elements, world) =

(* HELPERS *)

  (* Raise an exception if the given list has a duplicate *)
  let report_duplicate exceptf list =
    let rec helper = function
      n1 :: n2 :: _ when n1 = n2 -> raise (Failure (exceptf n1))
      | _ :: t -> helper t
      | [] -> ()
      in helper (List.sort compare list)
  in

  (* get variable name *)
  let varName = function (_, n, _) -> n in
```

```
  (* get bind name *)
  let bindName = function (_, n) -> n in

  (* Raise an exception of the given rvalue type cannot be assigned to
       the given lvalue type *)
  let check_assign lvaluet rvaluet err =
    if lvaluet == rvaluet then lvaluet else raise err
  in

  let string_of_typ = function
    Int -> "int"
    | Float -> "float"
    | Bool -> "bool"
    | Void -> "void"
    | Pair -> "pair"
    | Color -> "color"
    | String -> "string"
  in

  let rec string_of_expr = function
    ILiteral(l) -> string_of_int l
  | FLiteral(l) -> string_of_float l
  | BLiteral(true) -> "true"
  | BLiteral(false) -> "false"
  | SLiteral(s) -> s
  | Pr(x,y) -> "(" ^ string_of_expr x ^ "," ^ string_of_expr y ^ ")"
  | Cr(c) -> string_of_expr c
  | Id(s) -> s
  | Binop(e1, o, e2) ->
    string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2
  | Unop(o, e) -> string_of_uop o ^ string_of_expr e
  | Assign(v, e) -> string_of_expr v ^ " = " ^ string_of_expr e
  | Call(f, el) ->
    f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"
  | PAccess(s1,s2,e) -> s1 ^ "." ^ s2 ^ "." ^ e
  | CAccess(s1,s2) -> s1 ^ "." ^ s2
  | Keypress(e) -> "key_press(" ^ string_of_expr e ^ ")"
  | Noexpr -> ""

  in

  (**** Checking Global Variables ****)
  report_duplicate (fun n -> "duplicate global " ^ n) (List.map
varName globals);

  (* Type of each variable (global or formal *)
   let symbol = List.fold_left (fun m (t, n, _) -> StringMap.add n t
m)
     StringMap.empty globals
   in

  (* CHECK FUNCTIONS *)
    report_duplicate (fun n -> "duplicate function " ^ n)
      (List.map (fun fd -> fd.fname) funcs);
```

```
   let built_in_decls =  StringMap.add "print"
    { typ = Int; fname = "print"; formals = [(Int, "x")];
      locals = []; body = [] }
      (StringMap.singleton "add"
    { typ = Int; fname = "add"; formals = [(Bool, "e") ; (Pair,
"pos")];
      locals = []; body = [] })
   in

  let function_decls = List.fold_left (fun m fd -> StringMap.add
fd.fname fd m)
                     built_in_decls funcs in

  let function_decl s = try StringMap.find s function_decls
           with Not_found -> raise (E.UnrecognizedFunction(s)) in


    (* return the type of an ID (check symbols map) *)
    let type_of_identifier s m =
        try StringMap.find s m
        with Not_found -> raise (E.UndeclaredId(s))
    in


   (* Return the type of an expression or throw an exception *)
     let rec expr m = function
      ILiteral _ -> Int
      | BLiteral _ -> Bool
      | FLiteral _ -> Float
      | SLiteral _ -> Color
      | Id s -> type_of_identifier s m
      | Binop(e1, op, e2) as e -> let t1 = expr m e1 and t2 = expr m
e2 in
       (match op with
         Add | Sub | Mult | Div when t1 = Int && t2 = Int -> Int
         |  Add | Sub | Mult | Div when t1 = Int && t2 = Float ->
Float
         |  Add | Sub | Mult | Div when t1 = Float && t2 = Int ->
Float
         |  Add | Sub | Mult | Div when t1 = Float && t2 = Float ->
Float
         | Equal | Neq when t1 = t2 -> Bool
         | Less | Leq | Greater | Geq when t1 = Int && t2 = Int ->
Bool
         | Less | Leq | Greater | Geq when t1 = Float && t2 = Float ->
Bool
         | Less | Leq | Greater | Geq when t1 = Float && t2 = Int ->
Bool
         | Less | Leq | Greater | Geq when t1 = Int && t2 = Float ->
Bool
         | And | Or when t1 = Bool && t2 = Bool -> Bool
         | _ -> raise (E.IllegalBinOp("<" ^ string_of_typ t1 ^ "> " ^
string_of_op op ^ " <" ^
```

```
         string_of_typ t2 ^ "> in " ^ string_of_expr e))
       )
     | Unop(op, e) as ex -> let t = expr m e in
      (match op with
         Neg when t = Int -> Int
          | Not when t = Bool -> Bool
          | _ -> raise (E.IllegalUnOp(string_of_uop op, string_of_typ
t ^ " in ",
           string_of_expr ex))
      )
     | Noexpr-> Void
     | Assign(var, e) as ex -> let lt = expr m var
                               and rt = expr m e in
       check_assign lt rt (Failure ("illegal assignment <" ^
string_of_typ lt ^
                           "> = <" ^ string_of_typ rt ^ "> in " ^
                           string_of_expr ex))
     | Cr s -> let c = expr m s in
       let c1 = string_of_typ c in
       if c1 = "color" then Color
       else raise (E.IncorrectColorType("Please enter a 6-digit hex
number for color"))
     | Pr(x,y) -> let x1 = expr m x and y1 = expr m y in
       if (x1 = Int && y1 = Int) then Pair
       else raise (E.IncorrectPairType("Please enter int inputs for
type pair"))
     | Call(fname, actuals) as call -> let fd = function_decl fname in
              if List.length actuals != List.length fd.formals then
                raise (Failure ("expecting " ^ string_of_int
                (List.length fd.formals) ^ " arguments in " ^
string_of_expr call))
                else
                let checkSameT t1 t2 = if t1 != t2 then raise (Failure
("Incorrect actual argument type in " ^ string_of_expr call)) in
                List.iter2 (fun (ft,_) e -> let et = expr m e in
checkSameT ft et ) fd.formals actuals;
                fd.typ
     | PAccess (p, props, n) ->
          if not(StringMap.mem p m) then raise (E.UndeclaredId(p))
          else
            if not(props = "pos" || props = "size")
            then raise (E.NonAccessibleProp(props))
            else
              if (n = "x" || n = "y") then Int
              else raise (Failure("Please make sure you are assigning
to x or y and that it is a type: <int>"));

     | CAccess (player, prop) ->
          if (StringMap.mem player m && prop = "color") then
type_of_identifier prop m
          else
             raise (E.UndeclaredId(player))
     | Keypress (c) -> let move = string_of_expr c in
```

```
          if not(move = "UP" || move = "DOWN" || move = "LEFT" || move
= "RIGHT")
          then raise (E.WrongKeyPress); String

     in

     let checkBools m e = if expr m e != Bool
            then raise (E.NonBooleanType)
            else ()
       in

     (* Verify a statement or throw an exception *)
     let rec stmt m = function
       Block sl -> let rec check_block = function
         [Return _ as s] -> stmt m s
           | Return _ :: _ -> raise (E.PassedReturn("nothing may
follow a return statement"))
           | Block sl :: ss -> check_block (sl @ ss)
           | s :: ss -> stmt m s ; check_block ss
           | [] -> ()
             in check_block sl
         | Expr e -> ignore (expr m e)
         | Return _ -> ()
         | If (c, p1, p2) -> checkBools m c; stmt m p1; stmt m p2
         | While (c, s) -> checkBools m c; stmt m s
         | For(e1, e2, e3, st1) -> ignore(expr m e1); ignore(expr m
e2); ignore(expr m e3); stmt m st1
         | Condition (s1, s2) -> stmt m s1; stmt m s2
         | New (_) -> ()
         | ECall (addev,_) -> if addev = "add_event" then ()
     in

  let check_function func =

  report_duplicate (fun n -> "duplicate formal " ^ n ^ " in " ^
func.fname)
    (List.map bindName func.formals);

  let symbol = List.fold_left (fun m (t, n) -> StringMap.add n t m)
  symbol func.formals
  in

  report_duplicate (fun n -> "duplicate local " ^ n ^ " in " ^
func.fname)
    (List.map varName func.locals);

  (* Type of each variable locals *)
  let symbol = List.fold_left (fun m (t, n, _) -> StringMap.add n t m)
  symbol func.locals
  in

  stmt symbol (Block func.body)

  in
```

```
  List.iter check_function funcs;


  (* build a map given a list of members *)
  let memMap members =
    List.fold_left (fun m (t, n, _) -> StringMap.add n t m)
StringMap.empty members
  in

  (* check if a given member type exists *)
  let exist s t m =
    if not(StringMap.mem s m) then raise (E.UndefinedId(s))
    else
    let myMem = StringMap.find s m in
    if myMem != t then raise (E.IncorrectArgumentType("expected: <" ^
string_of_typ t ^
                                "> found: <" ^ string_of_typ myMem ^
">"))
  in


  (* add variable to a map *)
let addVar m (t, n, _) = StringMap.add n t m in

(* check if types in var_decl match *)
let checkVars m = function
(t,_,e) -> let ty = expr m e in
if t != ty
  then raise (E.IncorrectType("expected type:<" ^ string_of_typ t ^ ">
not (" ^ string_of_expr e ^ ") of type: " ^ string_of_typ ty))
in

(* CHECK ELEMENTS *)
let check_elements el =
  (* check required properties *)
   let elMems = memMap el.e_properties in
        exist "color" Color elMems;
        exist "size" Pair elMems;

 report_duplicate (fun n -> "Duplicate variable <" ^ n ^ "> in your
element properties")
   (List.map varName el.e_properties);

 let symbol = List.fold_left addVar StringMap.empty el.e_properties in
   List.iter (checkVars symbol) el.e_properties;

in

List.iter check_elements elements;



(*  CHECK WORLD *)
  let check_world w =
```

```
    (* check required properties *)
    let wMems = memMap w.w_properties in
          exist "color" Color wMems;
          exist "size" Pair wMems;

    (* check for duplicate world properties *)
    report_duplicate (fun n -> "Duplicate variable <" ^ n ^ "> in your
world properties")
      (List.map varName w.w_properties);

    let symbol = List.fold_left addVar StringMap.empty w.w_properties
in
    List.iter (checkVars symbol) w.w_properties;

    report_duplicate (fun n -> "Duplicate local variable <" ^ n ^ ">
in your world")
      (List.map varName w.init_locals);

    let symbol = List.fold_left addVar symbol w.init_locals in
    List.iter (checkVars symbol) w.init_locals;

    (* check world body *)
    stmt symbol (Block w.init_body);

    in
    check_world world;
```

codegen.ml

```
(* Code generation: translate takes a semantically checked AST and
produces LLVM IR

LLVM tutorial: Make sure to read the OCaml version of the tutorial

http://llvm.org/docs/tutorial/index.html

Detailed documentation on the OCaml LLVM library:

http://llvm.moe/
http://llvm.moe/ocaml/

*)

module L = Llvm
module A = Ast
module E = Exceptions

module StringMap = Map.Make(String)


let translate (globals, funcs, events, elements, world) =
  let context = L.global_context () in
  let the_module = L.create_module context "Craft"
```

```
  and i32_t  = L.i32_type  context (*int*)
  and flt_t = L.float_type context
  (* and i8_t   = L.i8_type   context (*printf format string / 8 bit
pointer*) *)
  and void_t = L.void_type context
  and i1_t   = L.i1_type   context (*bool*)
  and str_t  = L.pointer_type (L.i8_type context) in
  let color_t = L.named_struct_type context "color_t" in
  L.struct_set_body color_t [|i32_t; i32_t; i32_t|] false;
  let pair_t = L.named_struct_type context "pair_t" in
  L.struct_set_body pair_t [|i32_t; i32_t|] false;
  let elem_t = L.named_struct_type context "elem_t" in
  L.struct_set_body elem_t [|str_t; pair_t; pair_t; str_t; i32_t;
i32_t|] false;
  let world_t = L.named_struct_type context "world_t" in
  L.struct_set_body world_t [|pair_t;str_t|] false;

  (* Global map of events *)
  let fill_event_map m event =
    StringMap.add (event.A.evname ^ "_event") event m in
  let events_helper_map = List.fold_left fill_event_map
StringMap.empty events in

  let ltype_of_typ = function
      A.Int -> i32_t
    | A.Float -> flt_t
    | A.Bool -> i1_t
    | A.Pair -> pair_t
    | A.Color -> color_t
    | A.String -> str_t
    | A.Void -> void_t
  in


  let add_e_t = L.function_type (L.void_type context) [|
(L.pointer_type elem_t) |] in
  let add_e = L.declare_function "add_element" add_e_t the_module in

  let world_func_t = L.function_type i32_t [||] in
  let world_func = L.declare_function "world" world_func_t the_module
in

  let init_world_func_type = L.function_type (L.void_type context) [|
L.pointer_type world_t|] in
  let init_world_func = L.declare_function "init_world"
init_world_func_type the_module in

  let add_event_func_type = L.function_type (L.void_type context) [|
L.pointer_type (L.function_type (L.void_type context) [||]) |] in
  let add_event_func = L.declare_function "addEventfn"
add_event_func_type the_module in

  let is_key_pressed_func_type = L.function_type i32_t [|str_t|] in
```

```
  let is_key_pressed_func = L.declare_function "isPressed"
is_key_pressed_func_type the_module in

  let move_func_type = L.function_type (L.void_type context) [|str_t;
str_t|] in
  let move_func = L.declare_function "move" move_func_type the_module
in

  (* Helper functions *)
  let get_var_expr var_name var_list =
    let func = fun (_,s,_) -> s = var_name in
      match List.filter func var_list with
        | (_,_,e) :: tl -> e     (*removed:  :: tl  *)
        | [] -> A.Noexpr
        (* | _ -> A.Noexpr *)
  in

  let rec string_of_expr = function
    | A.SLiteral(s) -> s
    | A.Cr(c) -> string_of_expr c
    | A.Keypress(s) -> string_of_expr s
    | A.Id (s) -> s
    | _ -> ""
  in

  let get_var_decl_value = function
      A.ILiteral i -> L.const_int i32_t i
    | A.BLiteral b -> L.const_int i1_t (if b then 1 else 0)
    | A.Noexpr    -> L.const_int i32_t 0
    | _ -> L.const_int i32_t 0
  in

  (* Construct code for an expression; return its value *)
  let rec expr builder map = function
      A.ILiteral i -> L.const_int i32_t i
    | A.FLiteral f -> L.const_float flt_t f
    | A.SLiteral s -> L.const_string context s
    | A.BLiteral b -> L.const_int i1_t (if b then 1 else 0)
    | A.Noexpr -> L.const_int i32_t 0
    | A.Keypress s -> (expr builder map s)
    | A.Id s -> L.build_load (StringMap.find s map) s builder

    | A.Binop (e1, op, e2) ->
      let e1' = expr builder map e1
      and e2' = expr builder map e2 in
      (match op with
        A.Add     -> L.build_add
      | A.Sub     -> L.build_sub
      | A.Mult    -> L.build_mul
      | A.Div     -> L.build_sdiv
      | A.And     -> L.build_and
      | A.Or      -> L.build_or
      | A.Equal   -> L.build_icmp L.Icmp.Eq
      | A.Neq     -> L.build_icmp L.Icmp.Ne
```

```
      | A.Less    -> L.build_icmp L.Icmp.Slt
      | A.Leq     -> L.build_icmp L.Icmp.Sle
      | A.Greater -> L.build_icmp L.Icmp.Sgt
      | A.Geq     -> L.build_icmp L.Icmp.Sge
      ) e1' e2' "tmp" builder

    | A.Unop (op, e) ->
      let e' = expr builder map e in
      (match op with
        A.Neg     -> L.build_neg
      | A.Not     -> L.build_not
      ) e' "tmp" builder

    | A.Cr (e) ->
      let e' = expr builder map e in
      let cr_ptr = L.build_alloca color_t "tmp" builder in
      let hex_ptr = L.build_struct_gep cr_ptr 0 "hex" builder in
      ignore (L.build_store e' hex_ptr builder);
      L.build_load cr_ptr "c" builder

    | A.Pr (e1, e2) ->
      let e1' = expr builder map e1 in
      let e2' = expr builder map e2 in
      let pr_ptr = L.build_alloca pair_t "tmp" builder in
      let x_ptr = L.build_struct_gep pr_ptr 0 "x" builder in
      ignore (L.build_store e1' x_ptr builder);
      let y_ptr = L.build_struct_gep pr_ptr 1 "y" builder in
      ignore (L.build_store e2' y_ptr builder);
      L.build_load pr_ptr "p" builder

    | A.Assign (e1, e2) ->
      let e' = expr builder map e2 in (* should bring back calculated
value *)

      (match e1 with
        |A.Id (s) -> L.build_store e' (StringMap.find s map) builder
        |A.PAccess (s1,s2,s3) ->
          (match s2 with
            | "pos" ->
              (match s3 with
                | "x" ->   let elem_ptr = StringMap.find (s1 ^
"_element") map in
                           let pos_ptr = L.build_struct_gep elem_ptr
2 ("elem5_pos_ptr") builder in
                             let x_ptr = L.build_struct_gep pos_ptr 0
("x_ptr") builder in
                               L.build_store e' x_ptr builder;
                | "y" ->   let elem_ptr = StringMap.find (s1 ^
"_element") map in
                           let pos_ptr = L.build_struct_gep elem_ptr
2 ("elem6_pos_ptr") builder in
                             let y_ptr = L.build_struct_gep pos_ptr 1
("y_ptr") builder in
                               L.build_store e' y_ptr builder;
```

```
                | _ -> L.const_int i32_t 0
              )
            | "size" ->
              (match s3 with
                  | "x" -> let elem_ptr = StringMap.find (s1 ^
"_element") map in
                           let size_ptr = L.build_struct_gep elem_ptr
2 ("elem7_size_ptr") builder in
                             let x_ptr = L.build_struct_gep size_ptr
0 ("x_ptr") builder in
                               L.build_store e' x_ptr builder;
                  | "y" -> let elem_ptr = StringMap.find (s1 ^
"_element") map in
                           let size_ptr = L.build_struct_gep elem_ptr
2 ("elem8_size_ptr") builder in
                             let y_ptr = L.build_struct_gep size_ptr
1 ("y_ptr") builder in
                               L.build_store e' y_ptr builder;
                   | _ -> L.const_int i32_t 0
              )
            | _ ->  L.const_int i32_t 0
          )

        |  _ -> L.const_int i32_t 0
      )
    | A.PAccess (s1,s2,s3) ->
      (match s3 with
        | "x" ->
          (match s2 with
            | "pos" -> let elem_ptr = StringMap.find (s1 ^ "_element")
map in
                       let pos_ptr = L.build_struct_gep elem_ptr 2
("elem1_pos_ptr") builder in
                         let x_ptr = L.build_struct_gep pos_ptr 0
("x_ptr") builder in
                           L.build_load x_ptr "x" builder (*return
the x value in llvm type*)

            | "size" -> let elem_ptr = StringMap.find (s1 ^
"_element") map in
                         let size_ptr = L.build_struct_gep elem_ptr 2
("elem2_size_ptr") builder in
                           let x_ptr = L.build_struct_gep size_ptr 0
("x_ptr") builder in
                             L.build_load x_ptr "x" builder
            | _ -> L.const_int i32_t 0
          )
        | "y" ->
          (match s2 with
            | "pos" -> let elem_ptr = StringMap.find (s1 ^ "_element")
map in
                       let pos_ptr = L.build_struct_gep elem_ptr 2
("elem3_pos_ptr") builder in
```

```
                            let y_ptr = L.build_struct_gep pos_ptr 1
("y_ptr") builder in
                              L.build_load y_ptr "y" builder (*return
the y value in llvm type*)
            | "size" -> let elem_ptr = StringMap.find (s1 ^
"_element") map in
                            let size_ptr = L.build_struct_gep elem_ptr 2
("elem4_size_ptr") builder in
                              let y_ptr = L.build_struct_gep size_ptr 1
("y_ptr") builder in
                                L.build_load y_ptr "y" builder
            | _ -> L.const_int i32_t 0
          )
        | _ -> L.const_int i32_t 0
      )
    | A.CAccess (_,_) -> L.const_int i32_t 0 (*to stop warning*)
    | A.Call (f, act) ->
      let func = StringMap.find f map in
      let actuals = List.rev (List.map (expr builder map) (List.rev
act)) in
      L.build_call func (Array.of_list actuals) f builder

  in

  (* Invoke "f builder" if the current block doesn't already have a
terminal (e.g., a branch). *)
  let add_terminal builder f =
    match L.block_terminator (L.insertion_block builder) with
      Some _ -> ()
    | None -> ignore (f builder)
  in

  (* Build the code for the given statement; return the builder for
the statement's successor *)
  let rec stmt builder main_func map = function
        A.Block sl -> List.fold_left (fun builder s -> stmt builder
main_func map s) builder sl
      | A.Expr e -> ignore (expr builder map e); builder
      | A.Return e -> ignore(L.build_ret (expr builder map e)
builder); builder
      | A.New elem ->
        let (e_typ, _, e_pos) = elem in

        let elem_name = (e_typ ^ "_element") in
        let build_elem_func_name = (elem_name ^ "_build") in
        let build_elem_func = StringMap.find build_elem_func_name map
in
        let elem_ptr = L.build_call build_elem_func [||] "test_run"
builder in

        (* add the new Element position *)
        let pos_ptr = L.build_struct_gep elem_ptr 2 (elem_name ^
"_pos_ptr") builder in
```

```
        ignore (L.build_store (expr builder map e_pos) pos_ptr
builder);

        (* Call add_element function *)
        ignore (L.build_call add_e [|elem_ptr|] "" builder);
        builder

      | A.ECall ("add_event", event_name) ->

        let event = StringMap.find (event_name ^ "_event")
events_helper_map in
        let condition = string_of_expr event.A.condition in (*ex:
"UP"*)

        let first_elem = List.hd event.A.eformals in (*it's a string.
ex player which is now the name*)

        let event_func_type = L.function_type (L.void_type context)
[||] in
        let event_func = L.define_function (condition ^ "_event_func")
event_func_type the_module in
        let event_builder = L.builder_at_end context (L.entry_block
event_func) in (*event_func runs a new basic block. trigggered by
fucntion pointer in C*)

        let cond_str_ptr = L.build_global_stringptr condition
("condition_str_ptr") event_builder in
        let elem_name_str_ptr = L.build_global_stringptr first_elem
(first_elem ^ "_str_ptr") event_builder in

        (*call the c function and get a yes or no*)
        let return_val = L.build_call is_key_pressed_func [|
cond_str_ptr|] "" event_builder in (*stored in new basic block*)

        let is_pressed_bb = L.append_block context ("pressed_" ^
condition) event_func in
        let merge_bb = L.append_block context ("merge_" ^ condition)
event_func in

        let is_pressed_builder = L.builder_at_end context
is_pressed_bb in

        ignore(L.build_call move_func [|
elem_name_str_ptr;cond_str_ptr|] "" is_pressed_builder); (*tell C to
move element*)
        ignore (L.build_br merge_bb is_pressed_builder);

        let compare_instruction = L.build_icmp L.Icmp.Eq return_val
(L.const_int i32_t 1) ("key_pressed_"^condition) event_builder in
        ignore(L.build_cond_br compare_instruction is_pressed_bb
merge_bb event_builder);

        let merge_builder = L.builder_at_end context merge_bb in
        ignore (L.build_ret_void merge_builder);
```

```
        ignore (L.build_call add_event_func [|event_func|] ""
builder); (*giving C the pointer and it will call the func pointer*)

        builder

    | A.ECall (_,_) -> builder
    | A.Condition (_,_) -> builder
    | A.If (predicate, then_stmt, else_stmt) ->
        let bool_val = expr builder map predicate in
        let merge_bb = L.append_block context "merge" main_func in

        let then_bb = L.append_block context "then" main_func in
        add_terminal (stmt (L.builder_at_end context then_bb)
main_func map then_stmt)
        (L.build_br merge_bb);

        let else_bb = L.append_block context "else" main_func in
        add_terminal (stmt (L.builder_at_end context else_bb)
main_func map else_stmt)
        (L.build_br merge_bb);

        ignore (L.build_cond_br bool_val then_bb else_bb builder);
        L.builder_at_end context merge_bb

    | A.While (predicate, body) ->
        let pred_bb = L.append_block context "while" main_func in
        ignore (L.build_br pred_bb builder);

        let body_bb = L.append_block context "while_body" main_func in
        add_terminal (stmt (L.builder_at_end context body_bb)
main_func map body)
        (L.build_br pred_bb);

        let pred_builder = L.builder_at_end context pred_bb in
        let bool_val = expr pred_builder map predicate in

        let merge_bb = L.append_block context "merge" main_func in
        ignore (L.build_cond_br bool_val body_bb merge_bb
pred_builder);
        L.builder_at_end context merge_bb

    | A.For (e1, e2, e3, body) -> stmt builder main_func map
      ( A.Block [A.Expr e1 ; A.While (e2, A.Block [body ; A.Expr
e3]) ] )

  in

  (* Declare each global variable; remember its value in a map *)
  let global_vars_map =
    let global_var m (_, n, e) =
      let e' = get_var_decl_value e in
      StringMap.add n (L.define_global n e' the_module) m in
    List.fold_left global_var StringMap.empty globals
```

```
  in

  let function_decls_map =
    let function_decl map fdecl =
      let name = fdecl.A.fname
      and formal_types = Array.of_list (List.map (fun (t,_) ->
ltype_of_typ t) fdecl.A.formals) in
      let ftype = L.function_type (ltype_of_typ fdecl.A.typ)
formal_types in
      let func = L.define_function name ftype the_module in
      StringMap.add name func map
    in
    List.fold_left function_decl global_vars_map funcs
  in

let functions_map =
  let build_function_body m fdecl =
    let the_function = StringMap.find fdecl.A.fname m in
    let func_builder = L.builder_at_end context (L.entry_block
the_function) in

    let map =
        let add_formal map (t,n) p = L.set_value_name n p;
          let local = L.build_alloca (ltype_of_typ t) n func_builder
in
          ignore (L.build_store p local func_builder);
          StringMap.add n local map
        in

        let add_local map (t,n,e) =
          let e' = get_var_decl_value e in
          let local_var = L.build_alloca (ltype_of_typ t) n
func_builder in
          ignore(L.build_store e' local_var func_builder);
          StringMap.add n local_var map
        in

        let map = List.fold_left2 add_formal m fdecl.A.formals
(Array.to_list (L.params the_function)) in
        List.fold_left add_local map fdecl.A.locals
      in
    ignore(stmt func_builder the_function map (A.Block fdecl.A.body));
(*resolve function body statements*)
    map(* build_function_body returns the filled map *)
  in
  List.fold_left build_function_body function_decls_map funcs
in



  (* let elements_map =  *)
  let build_element_body (m, builder) element =

    (* ex: player_element *)
```

```
    let elem_name = (element.A.ename ^ "_element") in

    (* function used by new keyword to create element and add to world
*)
    let build_elem_func_name = (elem_name ^ "_build") in
    let build_elem_func_type = L.function_type (L.pointer_type elem_t)
[||] in
    let build_elem_func = L.define_function build_elem_func_name
build_elem_func_type the_module in
    let new_builder = L.builder_at_end context (L.entry_block
build_elem_func) in

    (* Element struct pointer *)
    let elem_ptr = L.build_malloc elem_t (elem_name ^ "_ptr")
new_builder in

    (* Element name *)
    let elem_name_str_ptr = L.build_global_stringptr element.A.ename
(elem_name ^ "_id_str_ptr") new_builder in
    let elem_name_ptr = L.build_struct_gep elem_ptr 0 (elem_name ^
"_name_ptr") new_builder in
    ignore (L.build_store elem_name_str_ptr elem_name_ptr
new_builder);

    (* Element size *)
    let elem_size_ptr = L.build_struct_gep elem_ptr 1 (elem_name ^
"_size_ptr") new_builder in
    let size_expr = get_var_expr "size" element.A.e_properties in
    ignore (L.build_store (expr new_builder m size_expr) elem_size_ptr
new_builder);

    (* Element color *)
    let color_expr = get_var_expr "color" element.A.e_properties in
    let color_str = string_of_expr color_expr in
    let elem_color_str_ptr = L.build_global_stringptr color_str
(elem_name ^ "_color_str_ptr") new_builder in
    let color_ptr = L.build_struct_gep elem_ptr 3 (elem_name ^
"_color_ptr") new_builder in
    ignore (L.build_store elem_color_str_ptr color_ptr new_builder);

    (* Element direction *)
    let elem_direction_ptr = L.build_struct_gep elem_ptr 4 (elem_name
^ "_direction_ptr") new_builder in
    let direction_expr = get_var_expr "direction"
element.A.e_properties in
    ignore (L.build_store (expr new_builder m direction_expr)
elem_direction_ptr new_builder);

    (* Element speed *)
    let elem_speed_ptr = L.build_struct_gep elem_ptr 5 (elem_name ^
"_speed_ptr") new_builder in
    let speed_expr = get_var_expr "speed" element.A.e_properties in
    ignore (L.build_store (expr new_builder m speed_expr)
elem_speed_ptr new_builder);
```

```
    ignore (L.build_ret elem_ptr new_builder); (*the functions return
value*)

    let m = StringMap.add build_elem_func_name build_elem_func m in
(*add the function so can call it in "NEW"*)

    (StringMap.add (elem_name) elem_ptr m, builder)
  in

  (* CREATE WORLD *)
  let world_start_func world map main_func builder =

    let world_ptr = L.build_malloc world_t ("world_ptr") builder in

    (* World size struct and pointer *)
    let world_size_ptr = L.build_struct_gep world_ptr 0 ("size_ptr")
builder in
    let size_expr = get_var_expr "size" world.A.w_properties in
    ignore (L.build_store (expr builder map size_expr) world_size_ptr
builder);

    (* World color struct and pointer *)
    let color_expr = get_var_expr "color" world.A.w_properties in
    let color_str = string_of_expr color_expr in
    let world_color_str_ptr = L.build_global_stringptr color_str
"color_str_ptr" builder in
    let color_ptr = L.build_struct_gep world_ptr 1 "color_ptr" builder
in
    ignore (L.build_store world_color_str_ptr color_ptr builder);

    (* World locals*)
    let map =
      let add_local map (t,s,e) =
        let e' = expr builder map e in
        let local_var = L.build_alloca (ltype_of_typ t) s builder in
            ignore (L.build_store e' local_var builder);
            StringMap.add s local_var map
      in
      List.fold_left add_local map world.A.init_locals
    in

    (* World statements *)
    let world_stmt_list = world.A.init_body in
    let builder = List.fold_left (fun b s -> stmt b main_func map s)
builder world_stmt_list in

    (world_ptr, builder)
  in

  (* MAIN FUNCTION *)
  let main_func_type = L.function_type i32_t [||] in
  let main_func = L.define_function "main" main_func_type the_module
in
```

```
  let main_func_builder = L.builder_at_end context (L.entry_block
main_func) in

  let (main_map, main_func_builder) = List.fold_left (fun (m,b) e ->
build_element_body (m, b) e) (functions_map, main_func_builder)
elements in

  let (world_ptr, main_func_builder) = world_start_func world main_map
main_func main_func_builder in

  ignore (L.build_call init_world_func [|world_ptr|] ""
main_func_builder);
  ignore (L.build_call world_func [||] "" main_func_builder);
  ignore (L.build_ret (L.const_int i32_t 0) main_func_builder);

the_module
```

craft.ml

```
(* Top-level of Craft compiler: scan & parse the input,
   check the resulting AST, generate LLVM IR, and dump the module *)

open Printf

module StringMap = Map.Make(String)

type action = Ast | LLVM_IR | Compile

let _ =
  let action = ref Compile in
  let set_action a () = action := a in
  let speclist = [
    ("-a", Arg.Unit (set_action Ast), "Print the SAST");
    ("-l", Arg.Unit (set_action LLVM_IR), "Print the generated LLVM
IR");
    ("-c", Arg.Unit (set_action Compile),
      "Check and print the generated LLVM IR (default)");
  ] in
  let usage_msg = "usage: ./craft.native [-a|-l|-c] [file.crf]" in
  let channel = ref stdin in
  Arg.parse speclist (fun filename -> channel := open_in filename)
usage_msg;
  let lexbuf = Lexing.from_channel !channel in
  let ast = Parser.program Scanner.token lexbuf in
  Semant.check ast;
  match !action with
    Ast -> print_string (Ast.string_of_program ast)
  | LLVM_IR -> print_string (Llvm.string_of_llmodule
(Codegen.translate ast))
  | Compile -> let m = Codegen.translate ast in
    Llvm_analysis.assert_valid_module m;
    (* print_string (Llvm.string_of_llmodule m) *)
    let arg_index =
      if Array.length Sys.argv == 2 then
```

```
        1
      else
        2 in

    let crf_file_in = Sys.argv.(arg_index) in
    let index = String.rindex crf_file_in '.' in
    let file_name = String.sub crf_file_in 0 index in


    let channel_out = open_out (file_name ^ ".ll") in
    fprintf channel_out "%s\n" (Llvm.string_of_llmodule m);
    close_out channel_out;
    let ll_name = file_name ^ ".ll" in
    let s_name = file_name ^ ".s" in
    let exe_name = file_name ^ ".exe" in

    let command_1 = "llc " ^ ll_name ^ " > " ^ s_name in
    ignore (Sys.command command_1);
    let command_2 = "gcc -o " ^ exe_name ^ " " ^ s_name ^ " main.o -
I /usr/local/include -L/usr/local/lib -lSDL2 `pkg-config --cflags --
libs glib-2.0`" in
    ignore (Sys.command command_2);
```

Makefile

```
#Craft Makefile in progress

LDFLAGS = lSDL2
H_DIR = header_files
INC_DIR = include
LIB_DIR = /usr/local/lib
CFLAGS = -Wall -w `pkg-config --cflags --libs glib-2.0`

.PHONY : all
all : craft.native main.o #link c here?

.PHONY : craft.native
craft.native :
     ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis -cflags -w,
+a-4 \
          craft.native

# "make clean" removes all generated files
.PHONY : clean
clean :
     ocamlbuild -clean
     rm -rf testall.log *.diff craft scanner.ml parser.ml parser.mli
     #rm -rf printbig
     rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe *.err
     rm -rf ./test_suite/*.o ./test_suite/*.s ./test_suite/*.ll ./
test_suite/*.out ./test_suite/*.exe ./test_suite/*.err


OBJS = ast.cmx codegen.cmx parser.cmx scanner.cmx semant.cmx craft.cmx
```

```
craft : $(OBJS)
     ocamlfind ocamlopt -linkpkg -package llvm -package llvm.analysis
$(OBJS) -o craft

scanner.ml : scanner.mll
     ocamllex scanner.mll

parser.ml parser.mli : parser.mly
     ocamlyacc parser.mly

%.cmo : %.ml
     ocamlc -c $<

%.cmi : %.mli
     ocamlc -c $<

%.cmx : %.ml
     ocamlfind ocamlopt -c -package llvm $<


### Generated by "ocamldep *.ml *.mli" after building scanner.ml and
parser.ml
ast.cmo :
ast.cmx :
codegen.cmo : ast.cmo
codegen.cmx : ast.cmx
craft.cmo : semant.cmo scanner.cmo parser.cmi codegen.cmo ast.cmo
craft.cmx : semant.cmx scanner.cmx parser.cmx codegen.cmx ast.cmx
parser.cmo : ast.cmo parser.cmi
parser.cmx : ast.cmx parser.cmi
scanner.cmo : parser.cmi
scanner.cmx : parser.cmx
semant.cmo : ast.cmo
semant.cmx : ast.cmx
parser.cmi : ast.cmo
```

8.2 Runtime

main.c

```
#include "SDL2/SDL.h"
#include "main.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include  <sys/types.h>
#include <math.h>

//REF: http://lazyfoo.net/tutorials/SDL/04_key_presses/index.php
//REF: https://wiki.libsdl.org
//REF: http://gamedevgeek.com/tutorials/moving-sprites-with-sdl/
```

```
int speed = 3;

// Changes direction for elastic collisions of the element passed depending on the axis of
collision.
void reflection(struct element *e, int axis){
   if(axis == 2){
      e->direction = 180 - e->direction;
   }else{
      e->direction = 360 - e->direction;
   }
   refresh = 1;
}

// Returns 0 if 2 elements do not collide. Returns 1 ro 2 depending on the axis that the
elements collide on if they do.
int doElementsCollide(struct element *e1, struct element *e2){

   // printf("%s %s \n", e1->name, e2->name);
   if(e1->position.left > e2->position.left + e2->size.left ||
      e1->position.left + e1->size.left < e2->position.left ||
      e1->position.right > e2->position.right + e2->size.right ||
      e1->position.right + e1->size.right < e2->position.right){
       return 0;
   }

   // Calls reflection for elastic collisions in case of collision
   if((e1->position.left < e2->position.left + e2->size.left + 3)&&
      (e1->position.left +3 > e2->position.left + e2->size.left)){
         reflection(e1, 2);
         return 2;
   }
   if((e1->position.left + e1->size.left + 3 > e2->position.left)&&
      (e1->position.left + e1->size.left < e2->position.left+3)){
         reflection(e1, 2);
         return 2;
   }
   if((e1->position.right < e2->position.right + e2->size.right + 3) &&
     (e1->position.right +3 > e2->position.right + e2->size.right)){
         reflection(e1, 1);
         return 1;
   }
   if((e1->position.right + e1->size.right + 3 > e2->position.right) &&
      (e1->position.right + e1->size.right < e2->position.right + 3)){
         reflection(e1, 1);
         return 1;
      }

   return 0;
}

// Refresh object depending on inherent speed and direction specified in the struct
void moveSpeed(struct element *e){

   double radians = M_PI * (e->direction/180.0);
```

```
    e->position.left = e->position.left + round(e->speed*cos(radians));
    e->position.right = e->position.right + round(e->speed*sin(radians));

    GSList* iterator = NULL;

    // Loop through the elements to check if the future space is already occupied by something
    for (iterator = element_list; iterator; iterator = iterator->next){
        struct element *temp = (struct element*)iterator->data;
        if(strcmp(e->name, temp->name)==0){
            continue;
        }
        if(doElementsCollide(e, temp)){
            e->position.left = e->position.left - round(e->speed*cos(radians));
            e->position.right = e->position.right - round(e->speed*sin(radians));
        }

        refresh = 1;
    }
}

void moveUp(struct element *e){
    if(e->position.right + e->size.right + speed < SCREEN_HEIGHT){
        e->position.right = e->position.right + speed;

        GSList* iterator = NULL;
        for (iterator = element_list; iterator; iterator = iterator->next)
        {
            if(strcmp(e->name, ((struct element*)iterator->data)->name)==0){
                continue;
            }
            if(doElementsCollide(e, (struct element*)iterator->data)){
                e->position.right = e->position.right - speed;
                return;
            }
        }
    }
}

void moveDown(struct element *e){
    if(e->position.right - speed >=0){
         e->position.right = e->position.right - speed;

        GSList* iterator = NULL;
        for (iterator = element_list; iterator; iterator = iterator->next)
        {
            if(strcmp(e->name, ((struct element*)iterator->data)->name)==0){
                continue;
            }
            if(doElementsCollide(e, (struct element*)iterator->data)){
                e->position.right = e->position.right + speed;
                return;
            }
        }
    }
```

```
}

void moveLeft(struct element *e){
   if(e->position.left - speed >=0){
      e->position.left = e->position.left - speed;

      GSList* iterator = NULL;
      for (iterator = element_list; iterator; iterator = iterator->next)
      {
         if(strcmp(e->name, ((struct element*)iterator->data)->name)==0){
            continue;
         }
         if(doElementsCollide(e, (struct element*)iterator->data)){
            e->position.left = e->position.left + speed;
            return;
         }
      }
   }
}

void moveRight(struct element *e){
   if(e->position.left + e->size.left + speed < SCREEN_WIDTH){
      e->position.left = e->position.left + speed;

      GSList* iterator = NULL;
      for (iterator = element_list; iterator; iterator = iterator->next)
      {
         if(strcmp(e->name, ((struct element*)iterator->data)->name)==0){
            continue;
         }
         if(doElementsCollide(e, (struct element*)iterator->data)){
            e->position.left = e->position.left - speed;
            return;
         }
      }
   }
}

void move(char *name, char *direction){

   struct element *e = NULL;
   GSList* iterator = NULL;

   for (iterator = element_list; iterator; iterator = iterator->next)
   {
      e = (struct element*)iterator->data;

      if(strcmp(e->name, name)!=0){
         e = NULL;
      }else{
         break;
      }
   }
   if(e==NULL){
```

```
        return;
    }

    refresh = 1;
    if (strcmp(direction, "UP") == 0 || strcmp(direction, "SUP") == 0){
        moveUp(e);
    }else if(strcmp(direction, "DOWN") == 0 || strcmp(direction, "SDOWN") == 0){
        moveDown(e);
    }else if(strcmp(direction, "LEFT") == 0 || strcmp(direction, "SLEFT") == 0){
        moveLeft(e);
    }else if(strcmp(direction, "RIGHT") == 0 || strcmp(direction, "SRIGHT") == 0) {
        moveRight(e);
    }
}

void (*event_fn)();

// Adding the pointer for the callback function for an event to the event pointer list
void addEventfn(void (*a)){
    fn_list = g_slist_append(fn_list, a);
}

// returns the state of the key associated with the string passed to it
int isPressed(char *key){
    int keyId;

    if (strcmp(key, "DOWN") == 0){
        keyId = 82;
    }else if(strcmp(key, "UP") == 0){
         keyId = 81;
    }else if(strcmp(key, "LEFT") == 0){
         keyId = 80;
    }else if(strcmp(key, "RIGHT") == 0) {
         keyId = 79;
    }else if(strcmp(key, "SPACE") == 0){
        keyId = 44;
    }else if (strcmp(key, "SDOWN") == 0){
        keyId = 26;
    }else if(strcmp(key, "SUP") == 0){
         keyId = 22;
    }else if(strcmp(key, "SLEFT") == 0){
         keyId = 4;
    }else if(strcmp(key, "SRIGHT") == 0) {
         keyId = 7;
    }
    if(keystate == NULL){
        return 0;
    }

    return keystate[keyId];
}

// draw an element on the the screen
void render_element(struct element *e) {
```

```
    SDL_Rect rect;
    rect.x = e->position.left;
    rect.y = e->position.right;
    rect.w = e->size.left;
    rect.h = e->size.right;
    SDL_FillRect(gScreenSurface, &rect, (int)strtol(e->el_color, NULL, 16));
}

// intialiaze the world
void init_world(struct world *temp){
    w=temp;
    SCREEN_WIDTH = w->size.left;
    SCREEN_HEIGHT = w->size.right;
}

// add element to main element list
void add_element(struct element *e){
    element_list = g_slist_append(element_list, e);
}

// refresh the world
void refresh_world(){
    SDL_FillRect(gScreenSurface, NULL, (int)strtol(w->back_color, NULL, 16));
}

// delete and alament from the main list
struct element* delete_element(char *name){

    struct element *e = NULL;
    GSList* iterator = NULL;

    for (iterator = element_list; iterator; iterator = iterator->next)
    {
        e = (struct element*)iterator->data;
        if(strcmp(e->name, name)!=0){
            e = NULL;
        }else{
            break;
        }
    }

    if(e !=NULL){
        element_list = g_slist_remove(element_list, iterator->data);
        refresh = 1;
        return e;
    }
    return NULL;

}

bool init()
{
    //Initialization flag
    bool success = true;
```

```
    //Initialize SDL
    if( SDL_Init( SDL_INIT_VIDEO ) < 0 )
    {
        printf( "SDL could not initialize! SDL_Error: %s\n", SDL_GetError() );
        success = false;
    }
    else
    {
        //Create window
        gWindow = SDL_CreateWindow( "Craft", SDL_WINDOWPOS_UNDEFINED,
SDL_WINDOWPOS_UNDEFINED, SCREEN_WIDTH, SCREEN_HEIGHT,
SDL_WINDOW_SHOWN );
        // gWindow = SDL_SetVideoMode(SCREEN_WIDTH, SCREEN_HEIGHT, 0, 0);
        if( gWindow == NULL )
        {
            printf( "Window could not be created! SDL_Error: %s\n", SDL_GetError() );
            success = false;
        }
        else
        {
            //Get window surface
            gScreenSurface = SDL_GetWindowSurface( gWindow );
        }
    }

    return success;
}

// Placeholder function to insert game logos, graphics, startup screens etc
bool loadMedia()
{
    //Loading success flag
    bool success = true;

    //Load splash image
    // gHelloWorld = SDL_LoadBMP( "hello_world.bmp" );
    // if( gHelloWorld == NULL )
    // {
    //     printf( "Unable to load image %s! SDL Error: %s\n",
"02_getting_an_image_on_the_screen/hello_world.bmp", SDL_GetError() );
    //     success = false;
    // }
    return success;
}

// Close and destroy the window
void close()
{
    //Destroy window
    printf("Exiting!!");
    SDL_DestroyWindow( gWindow );
    gWindow = NULL;
```

```
    //Quit SDL subsystems
    SDL_Quit();
}

int world( )
{
    //Start up SDL and create window
    if( !init() )
    {
        printf( "Failed to initialize!\n" );
    }
    else
    {
        //Load media
        if( !loadMedia() )
        {
            printf( "Failed to load media!\n" );
        }
        else
        {
            //Make the world of the size and backgorund
            SDL_FillRect(gScreenSurface, NULL, (int)strtol(w->back_color, NULL, 16));
            // Update the surface
            SDL_UpdateWindowSurface( gWindow );
            //Wait two seconds
            SDL_Delay( 2000 );
        }

        //Event handler
        SDL_Event e;

        bool quit = false;

        while(!quit){

            while( SDL_PollEvent (&e) != 0 ){

                //User requests quit
                if( e.type == SDL_QUIT ){
                    quit = true;
                }

                //User presses a key
                else if( e.type == SDL_KEYDOWN )
                {
                    //Select surfaces based on key press
                    switch( e.key.keysym.sym )
                    {
                        case SDLK_ESCAPE:

                        case SDLK_q:
                            quit = true;
                            break;
                    }
```

```
            }
        }

        GSList* iterator = NULL;
        // Iterate over the elements, update the positions of elements with defined speeds
        for (iterator = element_list; iterator; iterator = iterator->next)
        {
            render_element((struct element*)iterator->data);
        }

        // Iterate over the elements, re-render them.
        iterator = NULL;
        for (iterator = element_list; iterator; iterator = iterator->next)
        {
            moveSpeed((struct element*)iterator->data);
        }

        // Show changes on the window
        SDL_UpdateWindowSurface( gWindow );

        // Update the keymap
        keystate = SDL_GetKeyboardState(NULL);

        iterator = NULL;

        // Iterate over and call the callback functions
        for (iterator = fn_list; iterator; iterator = iterator->next)
        {
            void (*temp)() = (void *)iterator->data;
            temp();
        }

        // If the world needs to be refreshed, clean the slate
        if(refresh){
            refresh_world();
            refresh = 0;
        }

        // Delay to slow down the loop to a reasonable level
        SDL_Delay(5);
    }
  }

  return 0;
}
```

main.h

```
#include <stdbool.h>
#include <glib.h>

int SCREEN_WIDTH = 640;
int SCREEN_HEIGHT = 480;
```

```
const Uint8 *keystate = NULL;

// Element list
GSList* element_list;

// List of callback functions for events
GSList* fn_list = NULL;

//Starts up SDL and creates window
bool init();

//Loads media
bool loadMedia();

int isPressed(char* key);

//The window we'll be rendering to
SDL_Window* gWindow = NULL;

//The surface contained by the window
SDL_Surface *gScreenSurface = NULL;

// Flag to know when to refresh window
int refresh = 0;

// The world struct
struct world *w;

//Frees media and shuts down SDL
void close();

struct tuple{
    int left;
    int right;
};

struct color{
    int r;
    int g;
    int b;
};

struct element{
    char * name;
    struct tuple size;
    struct tuple position;
    char* el_color;
    // void (*event_fn)(struct element*);
    int direction;
    int speed;
};

struct world{
```

```
    struct tuple size;
    char* back_color;
};

// Changes direction for elastic collisions of the element passed depending on the axis of
collision.
void reflection(struct element *e, int axis);
// Returns 0 if 2 elements do not collide. Returns 1 ro 2 depending on the axis that the
elements collide on if they do.
int doElementsCollide(struct element *e1, struct element *e2);
// Refresh object depending on inherent speed and direction specified in the struct
void moveSpeed(struct element *e);

void moveUp(struct element *e);
void moveDown(struct element *e);
void moveLeft(struct element *e);
void moveRight(struct element *e);

// Wrapper function for all 4 move functions
void move(char *name, char *direction);

// Adding the pointer for the callback function for an event to the event pointer list
void addEventfn(void (*a));

// returns the state of the key associated with the string passed to it
int isPressed(char *key);

// draw an element on the the screen
void render_element(struct element *e);

// intialiaze the world
void init_world(struct world *temp);

// add element to main element list
void add_element(struct element *e);

// refresh the world
void refresh_world();

// delete and alament from the main list
struct element* delete_element(char *name);

// Setup and initialize the world
bool init();

// Placeholder function to insert game logos, graphics, startup screens etc
bool loadMedia();

// Fn to close window and close SDL
void close();

// Main world function
int world();
```

8.3 Pong Example

pong.crf

```
event move_right(paddle1) {
    condition = key_press("RIGHT");
    action {
      paddle1.pos.x = paddle1.pos.x + 5;
    }
}

event move_left(paddle1) {
    condition = key_press("LEFT");
    action {
      paddle1.pos.x = paddle1.pos.x - 5;
    }
}

event s_move_right(paddle2) {
    condition = key_press("SRIGHT");
    action {
      paddle2.pos.x = paddle2.pos.x + 5;
    }
}

event s_move_left(paddle2) {
    condition = key_press("SLEFT");
    action {
      paddle2.pos.x = paddle2.pos.x - 5;
    }
}

element ball {
    size = (14,14);
    color = "00ff00";
    direction = 70;
    speed = 2;
}

element wall_left {
    size = (10,600);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element wall_right {
    size = (10,600);
    color = "00ff00";
    direction = 0;
    speed = 0;
}
```

```
element wall_up {
    size = (600,15);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element wall_down {
    size = (600,10);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element paddle1{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element paddle2{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }

    element ball = new ball(100,100);
    element wall_left = new wall_left(0,0);
    element wall_right = new wall_right(585,0);
    element paddle1 = new paddle1(250,560);
    element paddle2 = new paddle2(250,40);
    add_event(move_right);
    add_event(move_left);
    add_event(s_move_left);
    add_event(s_move_right);
}
```

testall.sh

```
#!/bin/sh

# Regression testing script for MicroC
# Step through a list of files
#  Compile, run, and check the output of each expected-to-work test
#  Compile and check the error of each expected-to-fail test
```

```
# Path to the LLVM interpreter
LLI="lli"
#LLI="/usr/local/opt/llvm/bin/lli"

# Path to the LLVM compiler
LLC="llc"

# Path to the C compiler
CC="cc"

# Path to the microc compiler.  Usually "./microc.native"
# Try "_build/microc.native" if ocamlbuild was unable to create a
symbolic link.
CRAFT="./craft.native"
#MICROC="_build/microc.native"

# Set time limit for all operations
ulimit -t 30

globallog=testall.log
rm -f $globallog
error=0
globalerror=0

keep=0

Usage() {
    echo "Usage: testall.sh [options] [.crf files]"
    echo "-k    Keep intermediate files"
    echo "-h    Print this help"
    exit 1
}

SignalError() {
    if [ $error -eq 0 ] ; then
      echo "FAILED"
      error=1
    fi
    echo "  $1"
}

# Compare <outfile> <reffile> <difffile>
# Compares the outfile with reffile.  Differences, if any, written to
difffile
Compare() {
    generatedfiles="$generatedfiles $3"
    echo diff -b $1 $2 ">" $3 1>&2
    diff -b "$1" "$2" > "$3" 2>&1 || {
      SignalError "$1 differs"
      echo "FAILED $1 differs from $2" 1>&2
    }
}

# Run <args>
```

```
# Report the command, run it, and report any errors
Run() {
    echo $* 1>&2
    eval $* || {
      SignalError "$1 failed on $*"
      return 1
    }
}

# RunFail <args>
# Report the command, run it, and expect an error
RunFail() {
    echo $* 1>&2
    eval $* && {
      SignalError "failed: $* did not report an error"
      return 1
    }
    return 0
}

Check() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                              s/.crf//'`
    reffile=`echo $1 | sed 's/.crf$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""

    generatedfiles="$generatedfiles ${basename}.ll ${basename}.s $
{basename}.exe ${basename}.out" &&
    Run "$CRAFT" "$1" &&

    # Compare ${basename}.out ${reffile}.out ${basename}.diff

    # Report the status and clean up the generated files

    if [ $error -eq 0 ] ; then
      if [ $keep -eq 0 ] ; then
          rm -f $generatedfiles
      fi
      echo "OK"
      echo "###### SUCCESS" 1>&2
    else
      echo "###### FAILED" 1>&2
      globalerror=$error
    fi
}
```

```
CheckFail() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                              s/.crf//'`
    reffile=`echo $1 | sed 's/.crf$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""

    generatedfiles="$generatedfiles ${basename}.err ${basename}.diff"
&&
    RunFail "$CRAFT" "<" $1 "2>" "${basename}.err" ">>" $globallog &&
    Run "$CRAFT" "$1" &&
    # Compare ${basename}.err ${reffile}.err ${basename}.diff

    # Report the status and clean up the generated files

    if [ $error -eq 0 ] ; then
      if [ $keep -eq 0 ] ; then
          rm -f $generatedfiles
      fi
      echo "OK"
      echo "###### SUCCESS" 1>&2
    else
      echo "###### FAILED" 1>&2
      globalerror=$error
    fi
}

while getopts kdpsh c; do
    case $c in
      k) # Keep intermediate files
          keep=1
          ;;
      h) # Help
          Usage
          ;;
    esac
done

shift `expr $OPTIND - 1`

LLIFail() {
  echo "Could not find the LLVM interpreter \"$LLI\"."
  echo "Check your LLVM installation and/or modify the LLI variable in
testall.sh"
  exit 1
}
```

```
which "$LLI" >> $globallog || LLIFail



if [ $# -ge 1 ]
then
    files=$@
else
    files="test_suite/test-*.crf test_suite/fail-*.crf"

fi

for file in $files
do
    case $file in
      *test-*)
          Check $file 2>> $globallog
          ;;
      *fail-*)
          CheckFail $file 2>> $globallog
          ;;
      *)
          echo "unknown file type $file"
          globalerror=1
          ;;
    esac
done

exit $globalerror
```

8.4 Tests

8.4.1 Fails

fail-add-event.crf

```
element player {
    size = (10,10);
    color = "ffffff";
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    # Undefined event move
    add_event(move);
}
```

fail-assign1.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    int a = true; # fail assigning bool to int
}
```

fail-assign2.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    bool a = true; # fail assigning int to bool
    a = 5;
}
```

fail-assign3.crf

```
world {
    properties {
        size = (600,600);
        color = "000000";
    }
    int x = 3;
    color c = "aaaaaa";
    x = c; # cant assign a color to int
}
```

fail-binop.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    int a = 6 + (5,5); # fail with binary operator on different types
}
```

fail-binop2.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    int a = 6;
    bool b = true;
    a - b;
    a * b;
    a / b; # fail with binary operator on different types
```

```
}
```

fail-binop3.crf

```
world {
    properties {
        size = (600,600);
        color = "000000";
    }
    pair p = (3,3);
    pair p2 = (4,4);
    pair p3 = p / p2;  # cant divide pairs

}
```

fail-direction.crf

```
element paddle2{
    size = (60, 5);
    color = "00ff00";
    direction = true; # can only assign integers
    speed = 0;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }

}
```

fail-dup-globals.crf

```
int x = 3;
int x = 4; # fail on two identical ids

world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
}
```

fail-dup-locals.cf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    float a = 5.4;
    float a = 8.5; # fail on assigning same named float
```

```
}
```

fail-dup-prop.elem.crf

```
element player {
    size = (5,50);
    color = "4268f4";
    size = (30,3); # fail on duplicate sizes in element
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
}
```

fail-dup-prop.crf

```
world {
    properties {
        size = (1000,1000);
        color = "1a79e5";
        size = (5,5); # fail on duplicate sizes in world properties
    }
}
```

fail-elmt-no-clr.crf

```
element player1 {
    # fail with no color given to element
    size = (5,50);
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
}
```

fail-elmt-no-size.crf

```
element player1 {
    # fail with no size given to element
    color = "4268f4";
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
}
```

fail-for.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    # Unable to resolve x
    for (x ; x<3 ; x = x + 1) {
        y = y + 1;
    }
}
```

fail-func-decl-formals.crf

```
# missing arg types
def int foo(x,y) {
    return y;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }

}
```

fail-func-return.crf

```
# incorrect return type
def int foo(int x, bool y) {
    return y;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }
}
```

fail-if-else.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    # x is not defined
    if (x = 3) {
        x = 4;
    } else {
```

```
        x = 5;
    }
}
```

fail-new-stmt.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    # Undefined player element
    element player = new element(20,20);
}
```

fail-no-clr.crf

```
world {
    properties {
        size = (1000,1000);
        # fail with no color given
    }
}
```

fail-no-size.crf

```
world {
    properties {
        # fail with no size given
        color = "1a79e5";
    }
}
```

fail-return.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    int i = 3;
    return i;  # Should not return anything
    float a = 5.9;
}
```

fail-speed.crf

```
element paddle2{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = false;  # can only assign integers
}
```

```
world {
    properties {
        size = (600,600);
        color = "000000";
    }

}
```

fail-unary.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    bool x = true;
    x = -x; # Need ! on booleans

}
```

fail-while.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    # x is undefined
    while (x < 3) {
        x = x + 1;
    }
}
```

8.4.1 Passes

test-add-elements.crf

```
element player1 {
    size = (5,50);
    color = "4268f4";
}

element player2 {
    size = (30,30);
    color = "4268f4";
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
```

```
    element player1 = new player1(100,100);
    element player2 = new player2(300,300);
}
```

test-binop.crf

```
world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    int a = 6 + 5;
}
```

test-add-event.crf

```
event move(player) {
    condition = key_press("UP");
    action {
        player.pos.x = 5;
    }
}

element player {
    size = (10,10);
    color = "ffffff";
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    element player = new player(100,100);
    add_event(move);
}
```

test-comments.crf

```
#
# world {
#   properties {
#     size = (100,100);
#     color = "ffffff";
#   }
#
# }
#

world {
    properties {
        size = (100,100);
        color = "ffffff";
    }
```

```
}
```

test-fdecl-in-fdecl.crf

```
def int foo(int x) {
    return x;
}

def int foo1(int y) {
    return foo(y);
}

world {
  properties {
    size = (20,20);
    color = "ffffff";
  }
}
```

test-float-decl.crf

```
world {
  properties {
    size = (100,100);
    color = "ffffff";
  }
  float f = 3.4;
}
```

test-for-loop.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    int x = 3;
    int y = 4;
    for (x ; x<4; x = x + 1) {
        y = y + 1;
    }
}
```

test-func-binop.crf

```
def int func(int x,int y) {
    x = x + y;
    y = x / y;
    return x;
}

world {
    properties {
        size = (500,500);
```

```
        color = "f4f441";
    }
}
```

test-func-decl.crf

```
def int foo(int x) {
  return x;
}

world {
  properties {
    size = (20,20);
    color = "ffffff";
  }
}
```

test-func.crf

```
def int foo(int x) {
  return x;
}

world {
  properties {
    size = (20,20);
    color = "ffffff";
  }
  int a = foo(3);
}
```

test-function-assign.crf

```
def int add(int a, int b)
{
    return a + b;
}

element square {
    size = (50,50);
    color = "aaaaaa";
}

world {
    properties {
        size = (500,500);
        color = "42f4eb";
    }
    int x = 0;
    x = add(20,20);

    element square= new square(x,x);
}
```

test-function-using-globals.crf

```
int global = 3;

def int foo(int x) {
  return global;
}

world {
  properties {
    size = (20,20);
    color = "ffffff";
  }
}
```

test-global-binop.crf

```
int x = 3;
int y = x + 3;
int z = x + y;

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
}
```

test-global-vars.crf

```
int global_int = 3;
bool global_bool = true;
float global_float = 3.4;



world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
}
```

test-helloworld.crf

```
world {
  properties {
    size = (100,100);
    color = "ffffff";
    }
}
```

test-if-else.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    bool x = true;
    if (x) {
        x = false;
    } else {
        x = false;
    }
}
```

test-if.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    bool x = true;
    if (x) {
        x = false;
    }
}
```

test-int-decl.crf

```
world {
  properties {
    size = (100,100);
    color = "ffffff";
  }

  int x = 5; # Local var declaration in world

}
```

test-manual-move-left-right.crf

```
event move_right(paddle1) {
    condition = key_press("RIGHT");
    action {
      paddle1.pos.x = paddle1.pos.x + 5;
    }
}

event move_left(paddle1) {
    condition = key_press("LEFT");
    action {
      paddle1.pos.x = paddle1.pos.x - 5;
    }
}
```

```
element paddle1{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }
    # manually check left/right movement
    element paddle1 = new paddle1(250,560);
    add_event(move_right);
    add_event(move_left);
}
```

test-manual-move-up-down.crf

```
event move_up(paddle1) {
    condition = key_press("UP");
    action {
      paddle1.pos.x = paddle1.pos.x + 5;
    }
}

event move_down(paddle1) {
    condition = key_press("DOWN");
    action {
      paddle1.pos.x = paddle1.pos.x - 5;
    }
}

element paddle1{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }
    # manually check up/down movement
    element paddle1 = new paddle1(250,560);
    add_event(move_up);
    add_event(move_down);
}
```

test-manual-world-props.crf

```
world {
    properties {
        size = (500,500);  # 500 x 500 window size
        color = "48f442";  # Light green
    }
}
```

test-new-stmt.crf

```
element player {
    size = (10,10);
    color = "ffffff";
}

world {
    properties {
        size = (500,500);
        color = "f4f441";
    }
    element player = new player(20,20);
}
```

test-pong.crf

```
event move_right(paddle1) {
    condition = key_press("RIGHT");
    action {
      paddle1.pos.x = paddle1.pos.x + 5;
    }
}

event move_left(paddle1) {
    condition = key_press("LEFT");
    action {
      paddle1.pos.x = paddle1.pos.x - 5;
    }
}

event s_move_right(paddle2) {
    condition = key_press("SRIGHT");
    action {
      paddle2.pos.x = paddle2.pos.x + 5;
    }
}

event s_move_left(paddle2) {
    condition = key_press("SLEFT");
    action {
      paddle2.pos.x = paddle2.pos.x - 5;
    }
}

element ball {
    size = (14,14);
```

```
    color = "00ff00";
    direction = 70;
    speed = 2;
}

element wall_left {
    size = (10,600);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element wall_right {
    size = (10,600);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element wall_up {
    size = (600,15);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element wall_down {
    size = (600,10);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element paddle1{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

element paddle2{
    size = (60, 5);
    color = "00ff00";
    direction = 0;
    speed = 0;
}

world {
    properties {
        size = (600,600);
        color = "000000";
    }

    element ball = new ball(100,100);
```

```
    element wall_left = new wall_left(0,0);
    element wall_right = new wall_right(585,0);
    element paddle1 = new paddle1(250,560);
    element paddle2 = new paddle2(250,40);
    add_event(move_right);
    add_event(move_left);
    add_event(s_move_left);
    add_event(s_move_right);

}
```

test-unary.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    int x = 3;
    x = -x;
}
```

test-unary2.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    bool x = true;
    x = !x;
}
```

test-while.crf

```
world {
    properties {
        size = (500,200);
        color = "f4f441";
    }
    int x = 2;
    while (x < 3) {
        x = x + 1;
    }
}
```

test-world-multiple-vars.crf

```
world {
  properties {
    size = (100,100);
    color = "ffffff";
  }
  # Local vars declarations
```

```
  int x = 5;
  bool b = true;
  float f = 3.4;
}
```

8.5 Commit Log

commit d63d96542dc8f534dcace2dc5e2452b924866fa2
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 21:20:37 2017 -0500

cleanup

commit 5ffd8cc2b3da15336d4b35a063f8cbab03340446
Merge: bdd5d66 d254634
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 21:11:25 2017 -0500

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit bdd5d66f366a5196aeedad2adbcebf803444067f
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 21:11:16 2017 -0500

delete microc makefile

commit d254634716eda1bba108cea3e30413fbead6429c
Merge: d2d2132 0207511
Author: DTal <Dt2479@columbia.edu>
Date: Wed Dec 20 21:10:38 2017 -0500

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit d2d213273e84cc794be6ab202e89b1d7be723c35
Author: DTal <Dt2479@columbia.edu>
Date: Wed Dec 20 21:10:31 2017 -0500

semantmant

commit 0207511b7705d94cd3a7d3e6ab03a3c8141450d0
Merge: 237fff2 e642713
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 21:05:39 2017 -0500

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 237fff292a75bfd4fbca69f739449e1302527a23
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 21:05:32 2017 -0500

pkgconfig warning fix

commit e642713d3a02fe85c5f78d5900ab0321226d6b11
Merge: 4ba037f 655f519
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 21:00:42 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 4ba037f4582afb4790e6701ad76de05849a79990
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 21:00:33 2017 -0500

    final tests

commit 655f5198e2485e99e02b9e7e731e87b2eeeb2d89
Merge: eb49a9d f65d586
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 21:00:03 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit eb49a9d9e029815f192c26155743b00e2577056a
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 20:59:55 2017 -0500

    codegen clean. Probably final

commit f65d58697a29781b4422858cfcc13fa7d30de121
Merge: ac242ec aa0727d
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 20:54:27 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit ac242ec201417bd0aa007f7141446bdb4b5d52cd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 20:54:19 2017 -0500

    unary testing

commit aa0727d9e84a97eb22830b967fcd975ceddeea49
Author: DTal <Dt2479@columbia.edu>
Date: Wed Dec 20 20:53:58 2017 -0500

    fixed semant for functions

commit a1eef310d7e615642665f749ce4c262693c0aa8e
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 20:53:28 2017 -0500

    print statements removed from codgen and craft.ml

commit af41088fae43eb7e2e6513346dcba67bf31e68bf
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 20:45:34 2017 -0500

```
    function test

commit edfbff26623ba42dde1844d417025187b6954699
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Wed Dec 20 20:42:33 2017 -0500

    cleaned up codegen and fixed module bugs

commit 58a3044a0d93d36811134b2261790bf53e5a5b2d
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 20 20:41:32 2017 -0500

    added testing v1

commit b8a83d071ade48980e82b717e2f7d920b6d6b217
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 20 20:27:32 2017 -0500

    manual testing

commit 874ea305f6aeadfbdf1431355b268cc7f1c86741
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 20 20:11:37 2017 -0500

    cleared ast warnings

commit 11dd115a7b36470b8e184e6da5aa597692ec9cf9
Merge: 6141cc5 f0b1858
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 20 20:07:20 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 6141cc5c5375313d7f237d4cf033f36fa2cde248
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 20 20:07:12 2017 -0500

    even more tests

commit f0b185835824820a3eebb0f9036e73c59f670c9c
Merge: d9eb835 311c30c
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 20 19:58:25 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit d9eb83580418f130dc0ee301b30bfb8765110479
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 20 19:58:22 2017 -0500

    clean up c code

commit 311c30c4cf9d9bf12defd9fcc8414bbef68e6bc6
```

Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 19:55:34 2017 -0500

    more tests

commit 5f02c2f2a8317844d310266e542f567a6e6d23be
Merge: 4b14031 b7fb3f7
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 19:24:47 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 4b14031f9249b6718707871daf3c56644c87198a
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 19:24:36 2017 -0500

    added tests

commit b7fb3f7fddbadb69b253b7c62c38b915513e53aa
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 18:56:06 2017 -0500

    codegen clean

commit 502ffe4dd965dedc6af99bd838e9bbd3ec8a09dd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 18:53:59 2017 -0500

    added tests

commit bb07672239a6ac3c197f6ebe6ea84b7e0a5dca77
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 17:12:06 2017 -0500

    semant on

commit a8dd7ea3aae5e89224e6372deae3daa518fa3860
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 17:10:54 2017 -0500

    pong.crf

commit 12852c797d8d5c37ff5713918d225792189dc76d
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 17:02:38 2017 -0500

    pong demo

commit c01752d5c09f99a6c9944224d7cb886c05bfabf9
Merge: 7a571b5 93970ea
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 16:58:25 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 7a571b543e473147cc3c040233469591815ee924
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 16:58:20 2017 -0500

    final demo

commit 93970ea91d61a2214b0a44bf94df2bb825cb9ebd
Merge: 38a3cc1 9d215b9
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 16:39:05 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 38a3cc1efa8b74d4d56514f24289a089d020f4a1
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 16:38:54 2017 -0500

    codegen cleaning

commit 9d215b92d6d93fdeea9804c90ab602da4cf2656e
Merge: 26c3e35 ee0a671
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 16:02:09 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 26c3e35c6b99a17a5e84cf17fd44a02eae78913d
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 16:02:02 2017 -0500

    pong demo

commit ee0a671642cfc1bf98b1bf9c9ddfd12f39875dd4
Author: DTal <Dt2479@columbia.edu>
Date: Wed Dec 20 15:03:01 2017 -0500

    added for

commit b80bdaf063d0d48b1edd8f2576ef105e5e7bafbd
Merge: c4dfb3f 4c520e7
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 13:44:32 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit c4dfb3f1ac44e33c13e1221de0aaea1f8fed4931
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 13:44:26 2017 -0500

    movement except hori axes

commit 4c520e7faa836426b942e8c85321790372d6f32b
Author: DTal <Dt2479@columbia.edu>

Date: Wed Dec 20 13:40:42 2017 -0500

    fail rename

commit c3a9b95e9779ea69af86978058aeb0f942a03821
Author: DTal <Dt2479@columbia.edu>
Date: Wed Dec 20 13:39:48 2017 -0500

    rename for fails

commit 1074dd9f3c7636f8ded340d00e0ce74071e93cfe
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 13:31:10 2017 -0500

    makefile updates to clean test_sutie

commit 070c55de6814748e83d2f254b42fc426a2a0b636
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 13:09:58 2017 -0500

    for codegen

commit d6a286201ba095cf9e506915a1451c5e5537accd
Merge: 9cc3911 ea93514
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 13:05:14 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 9cc3911d5b34ce298820e7a556c1d129220cca21
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Dec 20 13:05:04 2017 -0500

    for loop modified

commit ea93514689606cf740a35149b0abf9bdd11345b4
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 13:03:54 2017 -0500

    while/if else codegen update

commit ad83abed4436aa50d7d2147a05f85121a36124a7
Merge: 69df97c 8e9d298
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 02:56:45 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 69df97c587946e3f60f170877fe62fd94d343e7d
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 02:56:39 2017 -0500

    codegen mods

commit 8e9d298250ae9fabf8cd0089170e246a37ee165a
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 02:49:38 2017 -0500

    testall partial

commit ca0d90a5b81adfdc52861a14be732cbbb34c0561
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 01:59:57 2017 -0500

    degrees to radians

commit c1a7e3f2681c3639f983ebbb504cce2ed37992ef
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 01:58:55 2017 -0500

    direction/speed

commit b24e627229cab1a736fb62616a199bd496416cea
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 01:48:10 2017 -0500

    add speed component

commit ffcb21d193d77bbccf153f202c5c7918f1384368
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 01:44:22 2017 -0500

    change element struct

commit 9524ee2e8b852c5b77608bd0ccd33ef82a59b8bb
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 20 01:16:23 2017 -0500

    checkpoint for movement

commit b8b18dac045d79ac4b7b1caca3967e888682ba77
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 01:09:39 2017 -0500

    codegen clean up

commit 847071d3962901804ea44c124da082e0cbec8c50
Merge: 640d304 4612e59
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 00:56:59 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 640d304d03dbaa61d3f3152b2a953c4efefa3b2a
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Dec 20 00:56:52 2017 -0500

    eliminated most warninings. Still shift reduce error in parser

commit 4612e59d867ba6d309c3af3c27d6f1213963180e
Merge: f74a1c0 947ebde
Author: DTal <Dt2479@columbia.edu>
Date: Tue Dec 19 23:28:35 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit f74a1c07186b5bedab625de570923679961b9b54
Author: DTal <Dt2479@columbia.edu>
Date: Tue Dec 19 23:28:31 2017 -0500

    Current final semant and added tests

commit 947ebde8f939d1aceaddb4dfd330f440d2861785
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 23:02:25 2017 -0500

    add collision check to movement

commit f9600befe627f0e752e97fd502b68237a12c4fac
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:45:49 2017 -0500

    main.j

commit ce20eb76c67df348b4b99fc84f1f01a53cb09365
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:42:19 2017 -0500

    list of events

commit ac33b1500b37a763a04d19831f627ce9da5eb2d8
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:31:10 2017 -0500

    fixing movement

commit 02f217829499d28491f0d51c915a089eccaae084
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:18:28 2017 -0500

    bracket

commit e9cb43d2c14c751a12c04597cdd961517b0e2a1f
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:16:40 2017 -0500

    change event fn parameter to voi

commit 7fe3982614d2e3707f2a4e2ee2b291d159720d68
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Dec 19 22:10:03 2017 -0500

```
    refresh

commit 1ba1c344acd956284dd1c155d78cc470e18d6870
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Tue Dec 19 21:58:44 2017 -0500

    move wrapper

commit 3d74ab1c5800e50a8e0122d72d6065bf4deff77b
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Tue Dec 19 21:27:52 2017 -0500

    movement fn

commit 4d7c814961f8b4fe974ce464c1921c81ac22c444
Merge: 3344822 9df4356
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Tue Dec 19 19:24:34 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 33448223497a0fb5f0174676faeff112a6ecb2e7
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Tue Dec 19 19:24:25 2017 -0500

    codgen progress

commit 9df435694cb39070da77904dc2c4fe2e95ae51ee
Author: DTal <Dt2479@columbia.edu>
Date:   Mon Dec 18 17:17:55 2017 -0500

    added test cases p1

commit 20ec4231ea59aae8948b1fc41b55810cccfb3a78
Author: DTal <Dt2479@columbia.edu>
Date:   Mon Dec 18 16:56:12 2017 -0500

    UD String type and semant cases

commit a6527be32f4dceb4fb231abb6083acf753ef95ed
Merge: cf39c73 3743fcd
Author: DTal <Dt2479@columbia.edu>
Date:   Mon Dec 18 02:23:12 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 3743fcdf8568be5d432c8a0b8489ee338d5d648f
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Mon Dec 18 02:22:30 2017 -0500

    reverse y axis

commit cf39c73cfd0bda69c3c54b739c19346a55fc3d2e
Merge: 403578d 05f1ee8
```

Author: DTal <Dt2479@columbia.edu>
Date: Mon Dec 18 02:22:13 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 403578d99a398d10f04341ec5973ddc3a8d200e3
Author: DTal <Dt2479@columbia.edu>
Date: Mon Dec 18 02:21:52 2017 -0500

    more semants
     Please enter the commit message for your changes. Lines starting

commit 05f1ee864737f5dc44a08b38e6547206ef30720b
Merge: 9d99045 015ce82
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Mon Dec 18 02:18:43 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 9d9904505c63e15adf268331c53ea66f2ba911c6
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Mon Dec 18 02:18:37 2017 -0500

    param of event_fn

commit 015ce82c9327eace45ee47729d1443bbe15a1387
Merge: 64496f4 6ff937e
Author: DTal <Dt2479@columbia.edu>
Date: Mon Dec 18 02:02:38 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 64496f454fcb0d7ee5403392afe6908d0d4d594f
Author: DTal <Dt2479@columbia.edu>
Date: Mon Dec 18 02:02:35 2017 -0500

    remade sem

commit 6ff937eefbf5400d8277ea2a0f85fd1e7fcb052d
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Dec 18 01:39:41 2017 -0500

    access fixed

commit c8bdd3ab009113159e432aea56117cf61e9701eb
Merge: 2d7afcf 038c363
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Mon Dec 18 01:17:26 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 2d7afcf7364401bf97962c70b028eedb3c5edef2
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Mon Dec 18 01:17:19 2017 -0500

```
    assign issue

commit 038c363c34cf5b9ae0191ed0bb60d4ecd0489c50
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Mon Dec 18 01:04:42 2017 -0500

    fix warnings and add testing

commit ba177cc6bd72dd64261e6b31a7852b3b7069dc08
Merge: fbdb512 1cac098
Author: DTal <Dt2479@columbia.edu>
Date:   Sun Dec 17 23:59:58 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit fbdb51274d450264d99fba68b84df8fe6be448d7
Author: DTal <Dt2479@columbia.edu>
Date:   Sun Dec 17 23:59:47 2017 -0500

    back two commits

commit 1cac098fbec58dccbb280c81c1362d0e666e72aa
Merge: 86b7da9 dc97485
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 23:58:57 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 86b7da95caa1d9b1a41a56b9face380c0ca648a3
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 23:58:47 2017 -0500

    readme modified

commit dc97485fcf3c8d5f275590c62947b9d5a30186bc
Author: DTal <Dt2479@columbia.edu>
Date:   Sun Dec 17 23:52:47 2017 -0500

    test semant

commit 76a2d590af199ee22df00ab7b683cbaa27f3a40d
Merge: bba0713 4d581ba
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 22:32:26 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit bba07130027fbfd1f4157394b282209bad8a864a
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 22:31:54 2017 -0500

    progress codegen
```

commit 4d581ba515e165580d4130f67072272c547e2de2
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 22:29:35 2017 -0500

    semant

commit b81686bf5c0bec3dc7ac90a189c7a7d6f93a0f25
Merge: 430f851 31496b8
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 22:05:51 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 430f851398107ef9bb6b21689d25c35ea6032253
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 22:05:48 2017 -0500

    sem for ab

commit 31496b84c4a932829c8d0ddc4a9a2f77e21076bd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Dec 17 20:05:07 2017 -0500

    minor fix for event formals

commit f1b69f8daa8aefd3d3b70ad95784e81dc3f7a7aa
Merge: 9c258db 9ba8e08
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Dec 17 20:02:04 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 9c258db4d72a9a9d978baabd234758d0849cb10d
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Dec 17 20:01:55 2017 -0500

    changed event constructors

commit 9ba8e084db3c391705631ce6ab362a24ec7c0772
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sun Dec 17 19:45:04 2017 -0500

    new implementation for new elements. Can only add 1 of each element with new command

commit f13f5363bda1d557c7e9d89a9f3934fa4508efae
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Dec 17 19:30:28 2017 -0500

    new statement changed

commit d328a97438df30fd0da2cdce3389d851c9016171
Merge: 230d568 a374f2b
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Dec 17 19:01:39 2017 -0500

```
    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 230d5681168ad1ada59baf68b25637ad6315b264
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 19:01:30 2017 -0500

    access expr modified

commit a374f2bf2dbd802935dc54aca9c90663db30245f
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 18:38:45 2017 -0500

    codgen mods

commit 5d713681c574c9b8b2eb794cfd5ac273de043d21
Merge: ee36105 2ec14a4
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 18:35:18 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit ee3610542b6742c1c4121c796082dc7e1a554489
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 18:35:08 2017 -0500

    moved Ecall to stmt

commit 2ec14a467795279c333c11b3d1038a2552632371
Merge: 86a0bee f2eab3d
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 18:29:05 2017 -0500

    chnages semant...

commit 86a0bee0cacde441a620018ae21394981423eaf2
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 18:27:19 2017 -0500

    codegen erroor gone

commit f2eab3d1013ea346af10d32642362abc4a29b3f7
Merge: 3397f70 0161add
Author: DTal <Dt2479@columbia.edu>
Date:   Sun Dec 17 18:13:22 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 3397f7009902d6f5f097c776567a126a2891b605
Author: DTal <Dt2479@columbia.edu>
Date:   Sun Dec 17 18:13:19 2017 -0500

    Semant to check
```

```
commit 0161adda017c72a9c7519e5171eb3320be734e06
Merge: a338022 0d4530a
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 17:59:19 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit a338022b65a375b268bb86765732381fd6962f0d
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 17:59:12 2017 -0500

    codegen progress

commit 0d4530a503c4c83463b69504ad927f3f1698ff13
Merge: 93a5f71 735a426
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 17:32:57 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 93a5f7169343047a945a5422074578369a70d32b
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Sun Dec 17 17:32:47 2017 -0500

    small change on event formals name

commit 735a4269c66ea21e589d64c4a43c5ad411e6c978
Merge: d9f0dcf e00e2c4
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 17:24:43 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit d9f0dcf12708d1fd836d0326e453ab2518d67533
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Sun Dec 17 17:24:33 2017 -0500

    new version of element handling

commit e00e2c4951713fd8d77a9c789f785b5df4522921
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Sun Dec 17 17:23:47 2017 -0500

    return type

commit 8b67b6c0ceeab47c83c849faf508c565b64835dc
Merge: 5e810c8 dd7d91f
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Sun Dec 17 17:09:41 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 5e810c87c805c0c842ae04fe6ebed7a31af45faa
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
```

Date: Sun Dec 17 17:09:32 2017 -0500

    check null

commit dd7d91f6bf57b537067ad9adbb8ef7578b58d0c3
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sun Dec 17 16:12:33 2017 -0500

    added elem store functions to codegen

commit 21105b315f7547adc232c540e98f399796c76292
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sun Dec 17 02:38:46 2017 -0500

    cathcing key press

commit 08ee11c0f1911154f807dd277b4717cf22aca445
Merge: 86426d8 9b0b3f8
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:44:33 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 86426d8ee53b95f02ce12dce33fa77ffc1773c72
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:44:26 2017 -0500

    refresh add

commit 9b0b3f8fbb42146ef846c25069873e4e45ba087e
Merge: dc3f014 f7521d6
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 01:27:16 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit dc3f0141e57fbad2410fa3d6e5c925ea26ea6c0c
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 01:27:14 2017 -0500

    exception fixes

commit f7521d64f3df71c66ebe0b6218086db5df925480
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:25:52 2017 -0500

    error

commit 7ad49529be0ffd2f8d3c1bf9218cc5060f94b1ad
Merge: 02af065 a7417f5
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:22:11 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 02af06523a764cf7474d391279930a80161f9679
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:22:04 2017 -0500

    return pointer to element

commit a7417f5d16630a4dbb5ebef9ef0452e82d6aad16
Merge: d8849c4 d4a4ad1
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 01:16:22 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit d8849c46220ff71bb2d61cf4d80d75eeef1d4804
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 01:16:18 2017 -0500

    update

commit d4a4ad1fc13c3d090801b5a301594cac3f905a15
Merge: 4d71f1e d65f991
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:14:39 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 4d71f1ef09fc1f4774e047e108a5ede2be4580ab
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sun Dec 17 01:14:34 2017 -0500

    call event in loop

commit d65f991e5144979a1e77209ed646d5767225834e
Author: DTal <Dt2479@columbia.edu>
Date: Sun Dec 17 00:55:11 2017 -0500

    Addition to Semant for functions

commit 7f307733edfc3c140d6464f7407b70f84cc27a68
Merge: 8140520 8e8305f
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 22:16:57 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 81405209829a3bb0ec8fdf3a90d36fbe5d752809
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 22:16:53 2017 -0500

    small semant bug fixed

commit 8e8305fa9477070e3f16c6b1026e21f3ffc6ed2e
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>

Date: Sat Dec 16 22:11:39 2017 -0500

    capital fix

commit 3ceb03298bbeb409936328de313f19cbfa70daa7
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 22:06:54 2017 -0500

    print key return

commit 6e583b1136dc256746fec1773b50d7261589f1a5
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:52:04 2017 -0500

    without delete

commit 4aafb735055cd65ca938e6e0c212c583da39d7dc
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:47:37 2017 -0500

    break

commit 7a0ad3f977b094f422ab412bfb3874f6812cf70b
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:42:02 2017 -0500

    print size

commit 36ed9087b782abb6ef4ea8e660b23ffb9058163d
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 21:22:59 2017 -0500

    Semant for abi

commit 7d890d91380fa42b3ec073ec34d9e7ff8d88bd92
Merge: 22bfeae fb67bf6
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:06:13 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit fb67bf6d2227955889de4d24328914b827fdb998
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 21:05:48 2017 -0500

    codegen. fixed funcs/globals

commit 22bfeae656b76174be0b24cb8128d9375906aa03
Merge: 786f6cb 96d9388
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:05:46 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 786f6cb755e6411a5b8f2ff5004c2c808a948daf
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 21:05:42 2017 -0500

    fix

commit 96d938858b404255bc35f4a7a64855e837f9cedb
Merge: 704a2a3 25a173e
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 19:47:12 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 704a2a36fe43c83816478ecd421dbd0d59680626
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 19:47:03 2017 -0500

    working on func body

commit 25a173ea828b7ec3a22e68d40b1353319efd7194
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 19:38:53 2017 -0500

    Semant update

commit e113e95a26937e22cfd55635f2496aea0fae518d
Merge: 2e95e0a b04ba72
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 18:50:01 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 2e95e0a3b1b3ad3b11584003fdabe4e9e1895e6f
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Dec 16 18:49:57 2017 -0500

    delete element

commit b04ba72ea39a25f7417ffaacfa40db9ea7327a09
Merge: 873c7db 526a7a7
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 18:43:54 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 873c7dbc83a0eec7af8119c531af73576ab7857c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 18:43:45 2017 -0500

    fixing func decls

commit 526a7a7ba883bf1f6b36ef3eed71f9a8715968c7
Merge: cfe6519 3ca0d06
Author: DTal <Dt2479@columbia.edu>

Date: Sat Dec 16 18:26:39 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit cfe6519aec0c1304a3d6a7a73cf212fd8a3cd258
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 18:26:27 2017 -0500

    Semant extras

commit 3ca0d060fd9d9d4c052099d6ce85f4ac4be85b4c
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 17:53:23 2017 -0500

    codegen clean

commit 6390d860d53afc64df4b8e2696cb6b6e4e35956a
Merge: b163726 85bf352
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 17:49:10 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit b1637269d4f68ca5c9ab0f062f69d0f2e75c7ffb
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 17:48:38 2017 -0500

    syntax errrrrrr

commit 85bf352e52f8ec79f6ad51c89a22c0c8b063b29a
Merge: 5f0a7b0 b13cf21
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 16:44:09 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 5f0a7b067ad5da8bc89cdc275bcd98ccc14102d7
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 16:43:59 2017 -0500

    access position fix

commit b13cf217f9195567c5c9e04b210e912c73a51d82
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Dec 16 16:43:54 2017 -0500

    codegen funcs

commit 54611f5eaf70321d862d98931a4c850d18c3260a
Merge: 81a2522 062fba9
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 14:26:55 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 81a25220ab911e53e4e7e835b3974722e3c39510
Author: DTal <Dt2479@columbia.edu>
Date: Sat Dec 16 14:26:51 2017 -0500

    Semant additions

commit 062fba95ba9e93a51160e4c85efddd7c538c03f5
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 13:53:14 2017 -0500

    fixed function decls

commit 8f3993f619bfbd9cc1a146ce02b848678a83431a
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Dec 16 13:03:31 2017 -0500

    function declarations changed

commit 2bbd673d4d9e59ca5e618950f423497cf5506288
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Fri Dec 15 19:09:46 2017 -0500

    function pointers working. C code printing out condition value for keypress

commit 2535686bc23a527118f39f5c1ea087a54d0f1894
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 18:51:20 2017 -0500

    null case

commit 45d656ee0d29790c41314cdd295417af53e46d17
Merge: 8e819b4 daddccb
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 18:37:17 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 8e819b4a9ed5e98d5b58c29399a1adfea2688198
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 18:37:13 2017 -0500

    change signature to int

commit daddccb6427d3659ab28e0b89e50e31119773796
Merge: a34612d ceb9ca1
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 17:08:08 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit a34612d10657d2fabd85fb59b839507efe1ee662
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 17:08:06 2017 -0500

```
    parser key_press

commit ceb9ca17da8d6798ee93c5cf1ef0f3cb6c9a0e3b
Merge: fe0b7e0 54fd12a
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Fri Dec 15 16:55:15 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit fe0b7e0f670570bb68f233465dadb444847ac25d
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Fri Dec 15 16:54:16 2017 -0500

    change return of isPressed to int

commit 54fd12a44c760c1151dc0ee49d1b0668b9111ce7
Author: DTal <Dt2479@columbia.edu>
Date:   Fri Dec 15 16:30:30 2017 -0500

    Semantics for elements

commit 255248a38e4546c5535f1d71cfa79a3b4afa99a5
Merge: aae370a d087c5c
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Fri Dec 15 16:23:16 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit aae370a233e6cd25fe89d41f97d9e6fa7a67f0c3
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Fri Dec 15 16:23:10 2017 -0500

    test fn

commit d087c5c0c1359d00d1bcb2512ba1d41efd9c0971
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Fri Dec 15 15:46:49 2017 -0500

    fixed main.h miss match

commit 03ce26d78d0238c98e030800a40d5a72c274de85
Merge: 45410f5 4a5d33e
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Fri Dec 15 15:42:24 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 45410f55ae88e6481a16ff6a203ce15a255c2054
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Fri Dec 15 15:42:18 2017 -0500

    trying to call function pinter
```

commit 4a5d33e3c2bc19ccacadeafd7949fa0910ce0097
Merge: aeed36c 8c8ae6b
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 15:41:37 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit aeed36ccfe513fafa4bfca753dc544187b2eba94
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 15:41:29 2017 -0500

    remove run

commit 8c8ae6bcf932c8086e5d5a0a14eeb316bd1a3fea
Merge: c241704 065f631
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 15:24:31 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit c241704b30ace561d8f8c78db9e7399ac334e38a
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 15:24:29 2017 -0500

    preperties naming fix

commit 065f63131ec39dc38c23951c94043d086852d040
Merge: 2d17a3c d47b780
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 15:23:56 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 2d17a3c623dff2b8f907279a3c09b76d0bc3df23
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 15:23:53 2017 -0500

    test and keybind

commit d47b780d35ac1a0c2651ba8cc29653c9dda53551
Merge: 8021f06 8fec31c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 14:35:57 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 8021f064e16466667f2678f31ca0fbe32b61269b
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 14:35:54 2017 -0500

    keypress expression

commit 8fec31c98d7fa148550e59238ea34609692d580b
Author: Martin Fagerhus <martin.fagerhus@gmail.com>

Date: Fri Dec 15 14:27:36 2017 -0500

    starting events

commit a23b20810126537dc528d1c767ff0ec8d0f7ace9
Merge: c7e1ad5 9a12d71
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 13:38:22 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit c7e1ad5201a734e0e2f2ef84e402d725ca390bbd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 13:38:19 2017 -0500

    ECall special function call

commit 9a12d71a6d760a1f7e3358e479cacfeb6375889d
Merge: 627e2d1 51a83cc
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 11:53:56 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 627e2d1b86825650d73ac5fdca97c80293ce1a76
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Fri Dec 15 11:53:49 2017 -0500

    add name to entity

commit 51a83cc60a69536c2f1a7341d5274b8f380eb3c2
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Dec 15 11:53:15 2017 -0500

    added ids to elements in codegen

commit 07719ae10dba114452a6504420caede69e78b04b
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Fri Dec 15 11:44:56 2017 -0500

    added events global string map

commit 9715b3be8587d5711036f0fded2426f079fb0110
Author: DTal <Dt2479@columbia.edu>
Date: Thu Dec 14 23:59:43 2017 -0500

    added tests for semantics

commit b11a2d5f28a552113530a1479b555704797d4c1e
Merge: 34d31e1 ef0b8b9
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 23:28:49 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 34d31e16ba7ed1ea1a1a89df404b0e8947794a64
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 23:28:47 2017 -0500

    events parsed properly

commit ef0b8b911aa11907062889b1c16d52229a2bf54a
Merge: 6d89d67 abec682
Author: DTal <Dt2479@columbia.edu>
Date: Thu Dec 14 21:58:32 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 6d89d676b944ca58986b8d3c60bbebf276348473
Author: DTal <Dt2479@columbia.edu>
Date: Thu Dec 14 21:58:00 2017 -0500

    semantic fixes and added tests

commit abec682800fc87c9326d909cd7134e3e892ff110
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Thu Dec 14 20:00:49 2017 -0500

    rendering element

commit 3377056c1d34f672531fe3b13f69f2f619d5abf2
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 19:51:07 2017 -0500

    new elements_map

commit 194ecc60bf97e70ee70d3e1582febc773646fe67
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 18:26:13 2017 -0500

    makefile edit

commit 17f3ce86e81f1dc54a2044b9d19923807beb3c77
Merge: 6486712 fc47716
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 17:14:16 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 6486712fed1938ab227fe4861d26bfc4ff4d35f5
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 17:14:12 2017 -0500

    issue with loading elements

commit fc47716b4f422d6f85737174650e4d3373f62c90
Merge: eda35b4 bd921ce
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>

Date: Thu Dec 14 17:09:26 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit eda35b445c40ba73678d27abba39a7db2a1fc22e
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Thu Dec 14 17:09:20 2017 -0500

    changed top level files

commit bd921ce1cd6b242bf85d29fa0a8e34b23b6438d3
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 17:05:14 2017 -0500

    add_element_function modified

commit 61e71ba8bbeea0938a7b2e75efcc9f8a7d2132ba
Merge: 2c42e4c afb0a9a
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Thu Dec 14 16:56:05 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 2c42e4c68222aa145de4399603419975a6ddaf05
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Thu Dec 14 16:55:58 2017 -0500

    create elemnt interface

commit afb0a9a03843386dfe009a10fb4702ea04880b5b
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 14 16:14:24 2017 -0500

    stored elements in map

commit 7777a4cdf2ecb165d2f4792266f31902a1c89793
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Thu Dec 14 15:40:33 2017 -0500

    working on elements

commit a99b4d192bc747bfc5a20f0745f6c5914fdf2889
Merge: 3daefa1 977291c
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Thu Dec 14 15:04:37 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit 3daefa1f849aca39977b14ace6bb5160bf94d972
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Thu Dec 14 15:04:21 2017 -0500

    fixed add_element bug-stringMapAdd

```
commit 977291ce97f22fe1c0a4eba82fe8486e93ddac73
Merge: dbf1fe0 b6007fb
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 14:49:52 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit dbf1fe0afa6b3a1b9ebf9d21cebfcd61cd296a55
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 14:49:49 2017 -0500

    test elements

commit b6007fb1925932923158ee23be94a0747f91c0d4
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Thu Dec 14 14:49:10 2017 -0500

    working on adding elements

commit 22357fc2a2f5647d0936aeedf7432f933643cf64
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 01:54:39 2017 -0500

    fixed hex color scanning

commit 95fe1d19557d4e8c82d9236e4d620f827fef8b5f
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Thu Dec 14 01:40:11 2017 -0500

    passing ast information correclty to c. Bug fixes

commit 2e2416dfb54d4543b4103bb5e8d61418d0656d0f
Merge: 44f889f 149a42e
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 01:25:46 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 44f889f11465c393a826fe7ab34fa55271fdfa8a
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 01:25:44 2017 -0500

    codegen world init modification

commit 149a42ed59c4ad4b07fce8a321e29168dd47a28e
Author: DTal <Dt2479@columbia.edu>
Date:   Thu Dec 14 01:22:14 2017 -0500

    semant checking for expr

commit 71f6542875a881edaa899480862a24d8e9750117
Author: DTal <Dt2479@columbia.edu>
Date:   Thu Dec 14 00:36:32 2017 -0500
```

```
    Semantics for world func

commit fc8f937885801107d6a24cdd0682ed1c76f4c7f5
Merge: e336a37 2de7835
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Thu Dec 14 00:09:52 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/plt_project

commit e336a37cb194e735838149c6e29122c5cd804b72
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date:   Thu Dec 14 00:09:21 2017 -0500

    now interacitng with c code from codegen

commit 2de78352e4f3368665f9f2a80f5acee37a544bdf
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Thu Dec 14 00:08:55 2017 -0500

    semantics modification

commit 56f6d036e45769b8775cdb5a02f8ec3a9f970945
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 22:43:01 2017 -0500

    c interface working with codegen

commit 2aaca45f876dc549f4cfe90584157642163f2aab
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 13 22:12:53 2017 -0500

    world struct changes

commit c1304bbdaf2bb07e83c22e06e60aac11e6493957
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 21:48:17 2017 -0500

    test codegen

commit 8e474a9589e450313d74742c7aafa5d3de10fe49
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 13 21:27:23 2017 -0500

    remove dangling conflict flag

commit fea06accb8cd604db99dc0b9a0da98fc26253ee1
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 13 21:18:04 2017 -0500

    c changes

commit 5686485e756cca95a6415238dfc3f8710bfd65ba
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 20:38:41 2017 -0500
```

```
    working world in codegen

commit 2c84c0f538ceb98dae827835439a43f67ce87a14
Merge: 0596448 f449888
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 17:02:11 2017 -0500

    merge changes

commit 0596448a2427510bcc5914780bf29a4fab210205
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 17:00:24 2017 -0500

    elements in codegen

commit f449888b9dbb63ff8c2e52697fba05e34473f853
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Wed Dec 13 16:26:50 2017 -0500

    add_element, init_world

commit c7b9ff7cdfe0d09eb36a82d9e517e14c2b759d02
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Wed Dec 13 14:54:56 2017 -0500

    codegen color pair modification

commit b830d1578e5b78d88c797b4adde1d14cd1655c3a
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Tue Dec 12 21:46:34 2017 -0500

    color and pair added to codegen expr

commit 479d814d014b0e00a862729261735c0cfc7ed361
Author: Roy Prigat <rp2719@columbia.edu>
Date:   Tue Dec 12 20:52:17 2017 -0500

    added expr to codegen

commit 76585184713fe582b05b7454e779ff493f9b696c
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Tue Dec 12 19:45:57 2017 -0500

    add types and element list create

commit 21d5d964b08ed4c38082f624287dec8ea1ee9f8c
Merge: 87d8c10 225bd97
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date:   Mon Dec 11 19:30:59 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 87d8c10f853cf89a524c57d75a14fd1c802be4b4
```

Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Mon Dec 11 19:30:53 2017 -0500

    reset codegen

commit 225bd979560024d9cfe2e7a31ea6c51bdfe66dc1
Merge: cee2d25 470b51d
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Dec 11 18:26:09 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit cee2d2556e88efc93394371a9a9ac3fcc7024537
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Dec 11 18:26:05 2017 -0500

    adding events

commit 470b51deb93bb7f31ef143b76488d31b612c5c7d
Merge: ed1fc3a 4e8a97d
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Mon Dec 11 18:24:17 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit ed1fc3aeaf625ab067d723f158e7ea1ad14ee053
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Mon Dec 11 18:24:10 2017 -0500

    header file refactor and elemnt add to list

commit 4e8a97d9997249a934e78accb54c3bb64e895353
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 7 18:34:40 2017 -0500

    dynamic assignments added

commit 139d50c9c5f7aed2a6da8178774f77bda1050bbd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Thu Dec 7 16:19:35 2017 -0500

    added element declaration to ast and parser

commit d72722052832264820e9da6f2af4e897c23d3aa8
Merge: 8401571 baa34f7
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 6 20:52:52 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 8401571036e17ddb14c47a53709566d9d54e59ef
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Dec 6 20:52:43 2017 -0500

interface gets cooler and cooler

commit baa34f73694f58b6f39cae1f85ab1887edbc26df
Merge: 8aa4a03 f715e6c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Dec 5 17:55:01 2017 -0500

Merged changes

commit 8aa4a035610321ef2ca002a7bbdeef48d4e8dd52
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Dec 5 17:49:54 2017 -0500

pretty printing elements

commit f715e6cd8d573d68f7d620b2e7fe252f456f6db3
Merge: 2642349 7a63f79
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Nov 29 21:43:31 2017 -0500

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit 26423494164677d73307621de7c4a4792e953baa
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Wed Nov 29 21:43:24 2017 -0500

fixed build, added glib, broken version of element rendering-trying options

commit 7a63f7999aed3cacf501f699b482cc68393f6a47
Author: DTal <Dt2479@columbia.edu>
Date: Wed Nov 29 19:53:25 2017 -0500

added func_decl, removed setvar, added a test file

commit 7caef8a3a58386c067caf20d65027ba8b485c39c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Nov 27 15:18:50 2017 -0500

test suite first tests edit

commit 6336e61fe3ea752e6206cfa780c9af7079e87b5a
Author: DTal <Dt2479@columbia.edu>
Date: Mon Nov 27 15:05:11 2017 -0500

Added test-suite folder

commit 013fe9f81ac1a7cc0642c4c0c3834eb571565a17
Merge: acdb007 b4d79cb
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 25 19:46:02 2017 -0500

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit acdb0077e14fdc4d892b6818e2aa9c5777e81727

Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 25 19:45:54 2017 -0500

    add struct, correct command for glib, so much wasted time ><

commit b4d79cb3cec4f3a2a618b5135dea797f4655a1c0
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Mon Nov 20 18:37:15 2017 -0500

    don't need shell script to compile executable anymore. All taken care of in craft.ml

commit afb8e5d63902a9f57d4425c615184ec93123de1f
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Mon Nov 20 14:38:14 2017 -0500

    hard coded codegen working for main directory, similar to simple version

commit d5ea1c26bc9434a4e37191c673326f434669425f
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Nov 18 19:13:13 2017 -0500

    updated gitignore

commit c1d285b72d65523727a7ed2caf8f9fa7f4d8d5a3
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Nov 18 19:10:13 2017 -0500

    printing ast, deleting test files

commit a881bcf97bf95b1f22d6e86a7bd437086b40d610
Merge: ec27b81 112c65c
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 18 18:19:32 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit ec27b81ab2e577098592a14b08ac09bfd633d0ad
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 18 18:18:31 2017 -0500

    add infinite run loop

commit 112c65c6f94a0265172bf8efa4e8e6aabebc68b0
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Nov 18 15:13:26 2017 -0500

    added simple_version directory with full copy of files. Some are stripped down. Manages to open window from c code

commit 893ac9f770db22c718aa00a48068389dba11dc02
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Nov 18 15:02:00 2017 -0500

    ast printing functions

commit d22e0987aba00330a8344f896917da2b907dea41
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sat Nov 18 14:37:34 2017 -0500

    starting semantics

commit 875f3a678599ce1955fa3665b2b9162d64037c4c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Wed Nov 15 16:24:09 2017 -0500

    Ready for helloworld

commit 1d577eb3ed6ce41774f984cbb6e87fdc0e53bdc0
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Nov 15 15:21:30 2017 -0500

    some Makefile edits

commit 502e21723eebc57365085264486ee568a37d6474
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Wed Nov 15 13:03:10 2017 -0500

    minor corrections to match parser and scanner

commit 94f01690c91986e8372d907e87f052c9af572cf8
Author: abhijeetmehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 11 20:51:41 2017 -0500

    Create readme.txt

commit 1d37078adadc73300f4bcccf89f8705dc832ffdc
Merge: c4b57e3 0376982
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 11 17:40:05 2017 -0500

    Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit c4b57e34c77c4282905ed43091d6da34d32481eb
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Sat Nov 11 17:39:54 2017 -0500

    first cut, c_interface

commit 03769826fe96c62b960b023bbec59fa99f55237f
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Nov 11 16:14:49 2017 -0500

    fixed syntax codegen.ml

commit e3391db5d90e709cc87b09379d092ad497b2dded
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Nov 11 16:12:00 2017 -0500

added our types to codegen.ml

commit 85a7374f081baded9509282a44c732a21ba1f70b
Author: Martin Fagerhus <martin.fagerhus@gmail.com>
Date: Sat Nov 11 15:34:11 2017 -0500

starting to structure codegen.ml

commit 146c37ce67075d8b5fface4c40647c06842516a5
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Nov 10 20:42:38 2017 -0500

helloworld edit

commit 92076a357b1a4acff5a41844ba716e62ba550317
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Nov 10 15:47:44 2017 -0500

helloworld v1

commit 31bbc33822956b45886c399665d97059db4de890
Author: Roy Prigat <rp2719@columbia.edu>
Date: Fri Nov 10 01:35:37 2017 -0500

parser helloworld

commit 512f80d4245e6b70015f1b970aaffe9167efca15
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 20:30:39 2017 -0500

Parser edit1

commit 6275e39dd8ebdf8ef5c5d7bac660b5c6273c1a68
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 15:17:05 2017 -0500

ast for helloworld

commit 25f2b114cba29d9f0b958841350c8ef8b2b54da4
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 14:20:41 2017 -0500

ast edit5

commit 8ed456b422a1922fceea78c06ec6dcba3a49470a
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 14:15:23 2017 -0500

ast edit4

commit e39ccd82699d0f35ac3a5efc110a141c4c5efaef
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Nov 7 14:11:45 2017 -0500

condition/action block adding

commit d425812bef66f9e06d9641f74252a87d7005fcd0
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 14:03:58 2017 -0500

ast edit3

commit 0c6430cbf77ba37c709dc6d418ad88e19989a3c3
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 13:51:17 2017 -0500

ast edit2

commit 6852535e894ade97b29a4938158ebeac04c72017
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Nov 7 13:48:05 2017 -0500

add collision op, nuke for loop

commit d5b32fe0e14ed7fd93e6150800800aedf9618acd
Author: Roy Prigat <rp2719@columbia.edu>
Date: Tue Nov 7 13:45:24 2017 -0500

ast edit

commit 1877cad6b379016b242dcb69e39dce9bac0b32e8
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Nov 7 13:17:07 2017 -0500

more scanner changes

commit 8d095ea0e13a20135a3a9f1c3b3d9d8b157fe333
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Nov 7 13:00:52 2017 -0500

illegal character in comment

commit eeaf97d1f396ca82944234915d03fc0843bc4321
Author: Abhijeet Mehrotra <abhijeet.meh@gmail.com>
Date: Tue Nov 7 12:59:07 2017 -0500

fix spacing scanner, add comment

commit 683472880bc716573ad83ad1c72b0931d4e24453
Merge: e67c7a2 8fc135e
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 21:39:26 2017 -0400

Merge branch 'master' of https://github.com/royprigat/PLT_Project

commit e67c7a221b44e8e572a2b957d391743e29a7ac33
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 21:39:23 2017 -0400

craft rename

commit 8fc135e3b554915dbe95d8630a4208369561cf12
Author: DTal621 <dt2479@columbia.edu>
Date: Mon Oct 30 21:27:10 2017 -0400

Add files via upload

microC test cases

commit bb257d55434d2b62a5e205b91ae0438ee8b9ee81
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 20:58:30 2017 -0400

Delete test.txt

commit 4cfa7a4b856190d11f3768a860cfd163cf8cc837
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 20:56:45 2017 -0400

Scanner edit

commit 3d760c7c0cbbd75f4aa0a3538e564b3c09d35741
Merge: 0479fa9 92cb7d4
Author: DTal <Dt2479@columbia.edu>
Date: Mon Oct 30 20:53:34 2017 -0400

addition of keywords

commit 0479fa990e1b71ab630b26f1aee269a1422444cd
Author: DTal <Dt2479@columbia.edu>
Date: Mon Oct 30 20:51:37 2017 -0400

Highlighted key words I havent implemented in scanner

commit 92cb7d49456ff8ff2a363ea7a02da340eed6851a
Merge: ee69b05 6fa50bc
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 20:35:47 2017 -0400

Scanner edit

commit ee69b051692c4983e72415d4149c8b58fe51603e
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 20:31:30 2017 -0400

Scanner edit

commit 6fa50bc1b287ac570316666c142882686f9cfdf5
Author: DTal <Dt2479@columbia.edu>
Date: Mon Oct 30 20:25:00 2017 -0400

comment change

commit af84eea2291008a083954b448368709a218d968e
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 20:07:38 2017 -0400

    Test

commit a6f066f07fcfc2f6f2356865f85de79292565160
Author: DTal <Dt2479@columbia.edu>
Date: Mon Oct 30 20:01:26 2017 -0400

    start of scanner update from microC

commit f08262b1598ace1d29ae87bb3c6fa81e72ade463
Author: DTal621 <dt2479@columbia.edu>
Date: Mon Oct 30 19:24:31 2017 -0400

    MicroC files

commit 6f3bf5a0a1436bde13322075c4b81a7b24daabe3
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 19:23:16 2017 -0400

    Remove calc templates

commit 13347a49cb6247e777e21d69e79a249dbde7f2f2
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 19:06:23 2017 -0400

    Professor Template Docs

commit 3647a72d24e4d3b66746db345232c419c1d8cf2f
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 19:02:27 2017 -0400

    dtal

commit c6c66054d4e51641043c5560f99ca2072d5c689d
Author: Roy Prigat <rp2719@columbia.edu>
Date: Mon Oct 30 17:24:56 2017 -0400

    Update README.md

commit a615244835f5ca9f3ae6aae4eea96b7ec21decf3
Author: DTal <Dt2479@columbia.edu>
Date: Sun Sep 10 16:52:02 2017 -0400

    sentence 2

commit 170e09670ca76a2dc0a0acac12b32e344cabcf49
Author: DTal <Dt2479@columbia.edu>
Date: Sun Sep 10 16:36:05 2017 -0400

    lik me balls

commit 52c1d45ce5275effbb012b3cf8c04ee400a1d9c9
Author: Daniel Tal <dtal@dyn-160-39-130-166.dyn.columbia.edu>
Date: Sun Sep 10 16:29:53 2017 -0400

    dtal test commit

commit 44a3527e3dd17a3558da7cb0602fc74a2ac8f84c
Author: Roy Prigat <rp2719@columbia.edu>
Date: Sun Sep 10 16:17:23 2017 -0400

    Initial commit