OGalaga

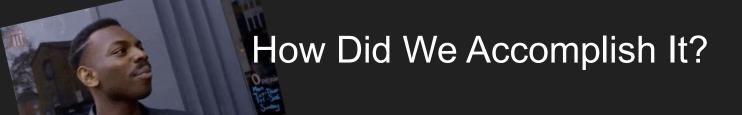## BúhoTech

Our Journey

PART ZERO: WHAT

Story

- Game development
- Proofs of possibility
- Flappy bird video
- ReasonML and Reprocessing
- Atom and Skype
- Google, off the table

How Did We Accomplish It?

Elements

STATE

Holds current values

SETUP

Initialize game state values

DRAW

Actions to be completed per frame. Passes an updated state to next frame.

RUN

Links IO events. Calls draw function every frame

PART ZERO.FIVE: REALIZATION

Remft Error


```
1   let keyPressed = ({shipX} as state, env) =>
2     Events.(
3       switch (Env.keyCode(env)) {
4       | Right
5       | D => {...state, rightPressed:true}
6       | Left
7       | A => {...state, leftPressed:true}
8       | _ => state
9       }
10    );
11  let keyReleased = ({shipX} as state, env) =>
12    Events.(
13      switch (Env.keyCode(env)) {
14      | Right
15      | D => {...state, rightPressed:false}
16      | Left
17      | A => {...state, leftPressed:false}
18      | _ => state
19      }
20    );
```

```
1   let keyPressed = ({shipX} as state, env) =>
2       Events.(
3           switch (Env.keyCode(env)) {
4               | Right
5               | D => {...state, rightPressed:true}
6               | Left
7               | A => {...state, leftPressed:true}
8               | _ => state
9           }
10      );
11      let keyReleased = ({shipX} as state, env) =>
12      Events.(
13          switch (Env.keyCode(env)) {
14          | Right
15          | D => {...state, rightPressed:false}
16          | Left
17          | A => {...state, leftPressed:false}
18          | _ => state
19      }
20  );
```

Story

"Feeling lost? Take a dream and convert it into small goals, then start taking the steps to hit those goals" - Dave Ramsey

..In other words..

Work piece by piece so when a piece break you know what to delete

CODE

PART 1: PROJECT STRUCTURE

```
1   open Reprocessing;
2
3   type stateT = {
4     /* Running Status */
5     gameHasStarted: bool,
6     gameWasStarted: bool,
7     exitStatus: bool,
8     playerDead: bool,
9     /* User Activities */
10    rightPressed: bool,
11    leftPressed: bool,
12    shotBool: bool,
13    score: int,
14    /* Actor Locations */
15    enemyShips: list((int, int)),
16    bulletPositions: list((int, int)),
17    starPositions: list((int, int)),
18    shipX: float,
19    /* Media */
20    enemyShipImage: imageT,
21    playerShipImage: imageT,
22    deadPlayerShipImage: imageT,
23    starImage: imageT,
24    bulletImage: imageT,
25    font: fontT,
26    backgroundMusic: soundT,
27    haveNotPlayedSongYet: bool,
28  };
```

```
31  let setup = env => {
32    Env.size(~width=600, ~height=800, env);
33    {
34      gameHasStarted: false,
35      gameWasStarted: false,
36      exitStatus: false,
37      playerDead: false,
38      rightPressed: false,
39      leftPressed: false,
40      shotBool: false,
41      score: 0,
42      enemyShips: [],
43      bulletPositions: [],
44      starPositions: [],
45      shipX: float_of_int(Env.width(env) / 2 - 30),
46      enemyShipImage: Draw.loadImage(~filename="assets/enemyShip.png", env),
47      playerShipImage: Draw.loadImage(~filename="assets/playerShip.png", env),
48      deadPlayerShipImage:
49        Draw.loadImage(~filename="assets/explosion.png", env),
50      starImage: Draw.loadImage(~filename="assets/playerBullet.png", env),
51      bulletImage: Draw.loadImage(~filename="assets/playerBullet.png", env),
52      font: Draw.loadFont(~filename="assets/fancy.fnt", ~isPixel=true, env),
53      backgroundMusic: Env.loadSound("assets/b.wav", env),
54      haveNotPlayedSongYet: true,
55    };
56  };
```

```
58    let draw =
59      (
60        {
61          gameHasStarted,
62          gameWasStarted,
63          exitStatus,
64          playerDead,
65          rightPressed,
66          leftPressed,
67          shotBool,
68          score,
69          enemyShips,
70          bulletPositions,
71          starPositions,
72          shipX,
73          enemyShipImage,
74          playerShipImage,
75          deadPlayerShipImage,
76          starImage,
77          bulletImage,
78          font,
79          backgroundMusic,
80          haveNotPlayedSongYet,
81        } as state,
82        env,
83      ) => {
```

```
317   let keyPressed =
318       (
319         {
320           shipX,
321           bulletPositions,
322           exitStatus,
323           gameHasStarted,
324           playerDead,
325           haveNotPlayedSongYet,
326         } as state,
327         env,
328       ) =>
329     Events.(
330       switch (Env.keyCode(env)) {
331       | Q => {...state, exitStatus: true}
332       | P =>···
350       | Right
351       | D => {...state, rightPressed: true}
352       | Left
353       | A => {...state, leftPressed: true}
354       | Space => {···
365       | _ => state
366       }
367     );
368
```

```
329    Events.(
330      switch (Env.keyCode(env)) {
331      | Q => {...state, exitStatus: true}
332      | P =>
333        playerDead
334          ? {
335            ...state,
336            playerDead: false,
337            gameHasStarted: true,
338            gameWasStarted: true,
339            exitStatus: false,
340            rightPressed: false,
341            leftPressed: false,
342            shotBool: false,
343            score: 0,
344            enemyShips: [],
345            bulletPositions: [],
346            starPositions: [],
347            shipX,
348          }
349          : {...state, gameHasStarted: !gameHasStarted, gameWasStarted: true}
350      | Right
351      | D => {...state, rightPressed: true}
352      | Left
353      | A => {...state, leftPressed: true}
354      | Space => {···
365      | _ => state
366      }
367    );
368
```

```
329        Events.(
330          switch (Env.keyCode(env)) {
331          | Q => {...state, exitStatus: true}
332  +       | P =>···
350          | Right
351          | D => {...state, rightPressed: true}
352          | Left
353          | A => {...state, leftPressed: true}
354          | Space => {
355              ...state,
356              shotBool: true,
357              bulletPositions:
358                gameHasStarted
359                  ? List.append(
360                      [(int_of_float(shipX +. 34.), 700)],
361                      bulletPositions,
362                    )
363                  : bulletPositions,
364          }
365          | _ => state
366          }
367        );
```

```
369   let keyReleased = (state, env) =>
370     Events.(
371       switch (Env.keyCode(env)) {
372       | Right
373       | D => {...state, rightPressed: false}
374       | Left
375       | A => {...state, leftPressed: false}
376       | _ => state
377       }
378     );
379
```

PART 2: DRAW FUNCTION

```
84    /* If the user hits 'Q' => exit window/game */
85    if (exitStatus) {
86      exit(0);
87    };
88
89    if (gameWasStarted && haveNotPlayedSongYet){
90      Env.playSound(backgroundMusic, ~volume=2.110, ~loop=true, env);
91    }
```

```
 93       /* Filter out ships that have collided with a bullet */
 94       let enemiesNotShot =
 95         List.filter(
 96           ((enemyX, enemyY)) =>
 97             !
 98               List.exists(
 99                 ((bulletX, bulletY)) =>
100                   Utils.intersectRectRect(
101                     (float_of_int(enemyX + 11), float_of_int(enemyY)),
102                     31.,
103                     40.,
104                     (float_of_int(bulletX), float_of_int(bulletY)),
105                     10.,
106                     10.,
107                   ),
108                 bulletPositions,
109               ),
110           enemyShips,
111         );
112
```

```
113    /* Filter out bullets that have collided with a ship */
114    let strayBullets =
115      List.filter(
116        ((bulletX, bulletY)) =>
117          !
118            List.exists(
119              ((enemyX, enemyY)) =>
120                Utils.intersectRectRect(
121                  (float_of_int(enemyX + 11), float_of_int(enemyY)),
122                  31.,
123                  40.,
124                  (float_of_int(bulletX), float_of_int(bulletY)),
125                  10.,
126                  10.,
127                ),
128              enemyShips,
129            ),
130        bulletPositions,
131      );
132
```

```
133    /*
134     * At this point, only ships that have been SHOT are DESTROYED (filter above)
135     * Therefore, we can give a point for every ship destroyed
136     */
137    let updatedScore =
138      List.length(enemyShips) > List.length(enemiesNotShot)
139        ? score + 1 : score;
140
```

```
141    /* filter out ships that hit player
142     * At this point in this frame, the only enemies left on screen are
143     * ships that have not gone out of bounds or been shot
144     * Any other collision would be between an enemy and player ship
145     */
146    let enemiesNotShotAndNotCollidedWith =
147      List.filter(
148        ((enemyX, enemyY)) =>
149          !
150            Utils.intersectRectRect(
151              (float_of_int(enemyX - 20), float_of_int(enemyY)),
152              31.,
153              40.,
154              (shipX, 700.00),
155              40.,
156              40.,
157            ),
158        enemiesNotShot,
159      );
160
```

```
161    /* previous step filtered ships that hit us
162     * If there are less ships since that check than a ship must have hit us
163     * Therefore, we can safely assert the status of the player
164     */
165    let isPlayerDead =
166      List.length(enemiesNotShot) > List.length(enemiesNotShotAndNotCollidedWith);
167
```

```
168     /* Now we can FILTER out ENEMY SHIPS that are out of bounds */
169     let enemiesStillOnScreen =
170       List.filter(((enemyX, enemyY)) => enemyY < 800, enemiesNotShotAndNotCollidedWith);
171

172     /* Now we can FILTER out BULLETS that are out of bounds */
173     let bulletsStillOnScreen =
174       List.filter(((xBullet, yBullet)) => yBullet > 0, strayBullets);
175

176     /* Now we can FILTER out STARS that are out of bounds */
177     let starPositions =
178       List.filter(((starX, starY)) => starY < 800, starPositions);
179
```

```
180    /* MOVE BULLETS UPWARD */
181    let bulletPositions =
182      List.map(((x, y)) => (x, y - 2), bulletsStillOnScreen);
183
184    /* MOVE STARS DOWNWARD */
185    let starPositions = List.map(((x, y)) => (x, y + 15), starPositions);
186
187    /* MOVE SHIPS DOWNWARD */
188    let enemiesWithUpdatedPositions =
189      List.map(((x, y)) => (x, y + 3), enemiesStillOnScreen);
190
```

```
191     /* respawn enemies */
192     let remainingEnemies =
193       List.length(enemyShips) < 12
194         ? List.append(
195             [
196               (
197                 Utils.random(50, Env.width(env) - 100),
198                 0 - Utils.random(28, 600),
199               ),
200             ],
201             enemiesWithUpdatedPositions,
202           )
203         : enemiesWithUpdatedPositions;
204
205     /* respawning stars */
206     let prettySky =
207       List.length(starPositions) < 52
208         ? List.append(
209             [
210               (
211                 Utils.random(-10, Env.width(env) + 10),
212                 0 - Utils.random(28, 600),
213               ),
214             ],
215             starPositions,
216           )
217         : starPositions;
218
```

```
219    /* Set new X COORDINATE for PLAYER */
220    let shipCurrentX =
221      rightPressed
222        ? shipX +. 4.2 > float_of_int(Env.width(env))
223          ? (-70.0) : shipX +. 4.20
224        : leftPressed
225          ? shipX -. 4.20 < (-70.0)
226            ? float_of_int(Env.width(env)) : shipX -. 4.20
227          : shipX;
228
```

```
229    /* If there are less enemies, after filtering out ships that have gone out of bounds
230     * then player failed to kill a ship. This is penalized with the less of a point
231     */
232    let pointsAfterOutOfBoundCheck =
233      enemiesStillOnScreen < enemiesNotShotAndNotCollidedWith ? updatedScore - 1 : updatedScore;
```

```
232    /* Paint background */
233    Draw.background(Utils.color(~r=0, ~g=15, ~b=25, ~a=255), env);
234
235    /* Draw stars */
236    List.iter(
237      item =>
238        Draw.pixel(
239          item,
240          Utils.color(
241            ~r=Utils.random(0, 255),
242            ~g=Utils.random(0, 255),
243            ~b=Utils.random(0, 255),
244            ~a=255,
245          ),
246          env,
247        ),
248      starPositions,
249    );
250
251    /* Draw enemies */
252    List.iter(item => Draw.image(enemyShipImage, ~pos=item, env), enemyShips);
253
254    /* Draw Player */
255    Draw.image(
256      playerDead ? deadPlayerShipImage : playerShipImage,
257      ~pos=(int_of_float(shipX), 700),
258      env,
259    );
260
```

```
264    /* Draw bullets */
265    List.iter(
266      ((x, y)) => Draw.image(bulletImage, ~pos=(x, y - 1), env),
267      bulletPositions,
268    );
269
270    /*
271     * Print SCORE if game has started
272     * Print START PROMPT if player has yet to start game
273     * Print RESUME PROMPT if game is pauseed
274     */
275    Draw.text(
276      ~font,
277      ~body=
278        playerDead
279          ? "GAME OVER - PRESS 'P' TO RESTART"
280          : gameHasStarted
281            ? string_of_int(score)
282            : gameWasStarted
283              ? "Press 'P' to resume or 'Q' to quit"
284              : "Press 'P' to start or 'Q' to quit",
285      ~pos=
286        playerDead
287          ? (Env.width(env) / 2 - 304, 40)
288          : gameHasStarted
289            ? (Env.width(env) / 2, 40) : (Env.width(env) / 2 - 260, 40),
290      env,
291    );
292
```

```
291    /* NEXT GAME STATE */
292    isPlayerDead
293      ? {...state, playerDead: true}
294      : gameHasStarted
295          ? {
296              ...state,
297              score: pointsAfterOutOfBoundCheck,
298              haveNotPlayedSongYet: false,
299              shotBool: false,
300              bulletPositions,
301              shipX: shipCurrentX,
302              enemyShips: remainingEnemies,
303              starPositions: prettySky,
304            }
305          /* IF GAME has yet to START or has been PAUSED
306           * If game PAUSED freeze PLAYER
307           * If game NEVER STARTED keep PLAYER CENTERED
308           */
309          : {
310              ...state,
311              shipX:
312                gameWasStarted ? shipX : float_of_int(Env.width(env) / 2 - 30),
313              gameHasStarted: false,
314            };
315    };
316
```

PART 3: RUN

```
380    run(~setup, ~draw, ~keyPressed, ~keyReleased, ());
```

PART 4: THE PRODUCT

Features

- Left and right movement wraps around screen
- Randomly spawned enemy ships
- Different enemy and player ship image
- Bullet & enemy destroyed on collision
- 1 point per enemy destroyed
- -1 point per enemy allowed to live
- Player destroyed on enemy collision
- Space themed background music
- Pause / Quit / Restart actions
- No Google

Future

- Enemy ships shoot back
- Player shooting speed reduced
- Different enemy ships
- Different weapons for player
- Ship selection for player
- Export to web or other accessible platforms
- Complete tutorial on YouTube coming soon

BATTLE ROYALE, MMORPG, VR, SPACE SHOOTER (WITH ESPORTS LEAGUE)

Trashed

- Avoid enemy ships adds 1 point
- Meteor elements
- Speed up animation
- Left Right movement animation
- Left Right enemy movement
- Enemy ships waves
- Shooting combos
- Shot sound effect
- Gun picks ups / power ups based on points

Two Minds are Better Than One

Peer programming through Atom as well as screen-sharing through

Skype, for every iteration of contributions.

Bouncing ideas off each other

OFu

NO GOOGLE.

ReasonML with Reprocessing have a VERY small community

Font colors are not supported. So we had to create one form a sprite sheet

Later on we found an application that does this

OThanks

Special thanks to:
My mom
My cat
My friends
Google
StackOverflow
Olexey Nikolaev
Benjamin San Souci
Avery Morin
kent

Press 'P' to start or 'Q' to quit

Bally