```cpp
#include "Collider.h"

Collider::Collider() {};

SDL_Rect* Collider::getColliderForStateStabThrowbottle(int curr_sprite, bool headLeft, SDL_Rect* appearance) {
    SDL_Rect* collider = NULL;
    if (headLeft == true) {
        switch (curr_sprite)
        {
        case 0:
            collider = new SDL_Rect();
            collider->x = appearance->x + 38;
            collider->y = appearance->y + appearance->h / 2;
            collider->w = appearance->w - 48;
            collider->h = appearance->h / 2;
            break;
        default:
            collider = new SDL_Rect();
            collider->x = appearance->x + 38;
            collider->y = appearance->y + appearance->h / 2;
            collider->w = appearance->w - 70;
            collider->h = appearance->h / 2;
            break;
        }
    }
    else if (headLeft == false) {
        switch (curr_sprite)
        {
        case 0:
            collider = new SDL_Rect();
            collider->x = appearance->x + 10;
            collider->y = appearance->y + appearance->h / 2;
            collider->w = appearance->w - 48;
            collider->h = appearance->h / 2;
            break;
        default:
            collider = new SDL_Rect();
            collider->x = appearance->x + 32;
            collider->y = appearance->y + appearance->h / 2;
            collider->w = appearance->w - 70;
            collider->h = appearance->h / 2;
            break;
        }
    }

    if (collider == NULL) {
        printf("Failed to return collider from class Collider.getColliderForStateStabThrowbottle(...)\n");
    }

    return collider;
};
```

```
52
53  SDL_Rect* Collider::getColliderForStateBlock(int curr_sprite, bool
      headLeft, SDL_Rect* appearance) {
54      SDL_Rect* collider = NULL;
55
56      if (headLeft == true) {
57          collider = new SDL_Rect();
58          collider->x = appearance->x + 43;
59          collider->y = appearance->y + appearance->h / 2;
60          collider->w = appearance->w - 86;
61          collider->h = appearance->h / 2;
62      }
63      else if (headLeft == false) {
64          collider = new SDL_Rect();
65          collider->x = appearance->x + 43;
66          collider->y = appearance->y + appearance->h / 2;
67          collider->w = appearance->w - 86;
68          collider->h = appearance->h / 2;
69      }
70
71      if (collider == NULL) {
72          printf("Failed to return Collider from class
              Collider.getColliderForStateBlock(...)\n");
73      }
74
75      return collider;
76  };
77
78  SDL_Rect* Collider::getColliderForStateIdleHurtWalk(int curr_sprite, bool
      headLeft, SDL_Rect* appearance) {
79      SDL_Rect* collider = NULL;
80
81      if (headLeft == true) {
82          collider = new SDL_Rect();
83          collider->x = appearance->x + 38;
84          collider->y = appearance->y + appearance->h / 2;
85          collider->w = appearance->w - 70;
86          collider->h = appearance->h / 2;
87      }
88      else if (headLeft == false) {
89          collider = new SDL_Rect();
90          collider->x = appearance->x + 32;
91          collider->y = appearance->y + appearance->h / 2;
92          collider->w = appearance->w - 70;
93          collider->h = appearance->h / 2;
94      }
95
96      if (collider == NULL) {
97          printf("Failed to return collider from class
              Collider.getColliderForStateIdleHurtWalk(...)\n");
98      }
99
100     return collider;
```

```
101 };
102
103 SDL_Rect* Collider::getColliderForStateJump(int curr_sprite, bool
      headLeft, SDL_Rect* appearance) {
104     SDL_Rect* collider = NULL;
105
106     if (headLeft == true) {
107         collider = getColliderForStateJumpHeadLeft(curr_sprite,
              appearance);
108     }
109     else if (headLeft == false) {
110         collider = getColliderForStateJumpHeadRight(curr_sprite,
              appearance);
111     }
112
113     if (collider == NULL) {
114         printf("Failed to return collider from class
              Collider.getColliderForStateJump(...)\n");
115     }
116
117     return collider;
118 };
119
120 SDL_Rect* Collider::getColliderForStateJumpHeadLeft(int curr_sprite,
      SDL_Rect* appearance) {
121     SDL_Rect* collider = NULL;
122
123     switch (curr_sprite)
124     {
125     case 0:
126         collider = getColliderForStateIdleHurtWalk
              (curr_sprite,true,appearance);
127         break;
128     case 1:
129         collider = getColliderForStateIdleHurtWalk(curr_sprite, true,
              appearance);
130         collider->y -= 5;
131         collider->h += 6;
132         break;
133     case 2:
134         collider = getColliderForStateIdleHurtWalk(curr_sprite, true,
              appearance);
135         collider->y -= 13;
136         collider->h += 10;
137         break;
138     case 3:
139         collider = getColliderForStateIdleHurtWalk(curr_sprite, true,
              appearance);
140         collider->y -= 14;
141         collider->h += 6;
142         break;
143     case 4:
144         collider = getColliderForStateIdleHurtWalk(curr_sprite, true,
```

```
            appearance);
145         collider->y -= 10;
146         break;
147     default:
148         printf("Failed to return collider from class
              Collider.getColliderForStateJumpHeadLeft(...). Invalid
              curr_sprite.\n");
149         break;
150     }
151
152     return collider;
153 };
154
155 SDL_Rect* Collider::getColliderForStateJumpHeadRight(int curr_sprite,
      SDL_Rect* appearance) {
156     SDL_Rect* collider = NULL;
157
158     switch (curr_sprite)
159     {
160     case 0:
161         collider = getColliderForStateIdleHurtWalk(curr_sprite, false,
              appearance);
162         break;
163     case 1:
164         collider = getColliderForStateIdleHurtWalk(curr_sprite, false,
              appearance);
165         collider->y -= 5;
166         collider->h += 6;
167         break;
168     case 2:
169         collider = getColliderForStateIdleHurtWalk(curr_sprite, false,
              appearance);
170         collider->y -= 13;
171         collider->h += 10;
172         break;
173     case 3:
174         collider = getColliderForStateIdleHurtWalk(curr_sprite, false,
              appearance);
175         collider->y -= 14;
176         collider->h += 6;
177         break;
178     case 4:
179         collider = getColliderForStateIdleHurtWalk(curr_sprite, false,
              appearance);
180         collider->y -= 10;
181         break;
182     default:
183         printf("Failed to return collider from class
              Collider.getColliderForStateJumpHeadLeft(...). Invalid
              curr_sprite.\n");
184         break;
185     }
186
```

```
187     return collider;
188 };
189 
190 SDL_Rect* Collider::getColliderForPlayerSwordHeadLeft(int curr_sprite,
      SDL_Rect* appearance){
191     SDL_Rect* collider = NULL;
192 
193     if (curr_sprite == 0) {
194         collider = new SDL_Rect();
195         collider->x = appearance->x + 91;
196         collider->y = appearance->y + 64 + 19;
197         collider->w = 30;
198         collider->h = 34;
199     }
200     else if ((curr_sprite == 1) || (curr_sprite == 2)) {
201         collider = new SDL_Rect();
202         collider->x = appearance->x;
203         collider->y = appearance->y + appearance->h - 17;
204         collider->w = 36;
205         collider->h = 15;
206     }
207     else if (curr_sprite == 3) {
208         collider = new SDL_Rect();
209         collider->x = appearance->x + 70;
210         collider->y = appearance->y + 75;
211         collider->w = 15;
212         collider->h = 36;
213     }
214     else if (curr_sprite == 4) {
215         collider = new SDL_Rect();
216         collider->x = appearance->x + 80;
217         collider->y = appearance->y + 72;
218         collider->w = 15;
219         collider->h = 36;
220     }
221 
222     if (collider == NULL) {
223         printf("Failed to return collider from class
              Collider.getColliderForPlayerSwordHeadLeft(...). Invalid
              curr_sprite.\n");
224     }
225 
226     return collider;
227 };
228 
229 SDL_Rect* Collider::getColliderForPlayerSwordHeadRight(int curr_sprite,
      SDL_Rect* appearance){
230     SDL_Rect* collider = NULL;
231 
232     if (curr_sprite == 0) {
233         collider = new SDL_Rect();
234         collider->x = appearance->x + 7;
235         collider->y = appearance->y + 64 + 19;
```

```
236         collider->w = 30;
237         collider->h = 34;
238     }
239     else if ((curr_sprite == 1) || (curr_sprite == 2)) {
240         collider = new SDL_Rect();
241         collider->x = appearance->x + 64 + 28;
242         collider->y = appearance->y + appearance->h - 17;
243         collider->w = 36;
244         collider->h = 15;
245     }
246     else if (curr_sprite == 3) {
247         collider = new SDL_Rect();
248         collider->x = appearance->x + 43;
249         collider->y = appearance->y + 75;
250         collider->w = 15;
251         collider->h = 36;
252     }
253     else if (curr_sprite == 4) {
254         collider = new SDL_Rect();
255         collider->x = appearance->x + 33;
256         collider->y = appearance->y + 73;
257         collider->w = 15;
258         collider->h = 36;
259     }
260
261     if (collider == NULL) {
262         printf("Failed to return collider from class
              Collider.getColliderForPlayerSwordHeadRight(...). Invalid
              curr_sprite.\n");
263     }
264
265     return collider;
266 };
267
268 //ACCESS
269 SDL_Rect Collider::getColliderForPlayer(int state, int curr_sprite, bool
      headLeft, SDL_Rect appearance) {
270     SDL_Rect* collider = NULL;
271
272     if ((state == Player::STAB) || (state == Player::THROWBOTTLE)) {
273         collider = getColliderForStateStabThrowbottle(curr_sprite,
              headLeft, &appearance);
274     }
275     else if ((state == Player::HURT) || (state == Player::IDLE) || (state
          == Player::WALK)) {
276         collider = getColliderForStateIdleHurtWalk(curr_sprite, headLeft,
              &appearance);
277     }
278     else if ((state == Player::BLOCK)) {
279         collider = getColliderForStateBlock(curr_sprite, headLeft,
              &appearance);
280     }
281     else if ((state == Player::JUMP)) {
```

```
282         collider = getColliderForStateJump(curr_sprite, headLeft,
              &appearance);
283     }
284
285     if (collider == NULL) {
286         printf("Failed to return Collider from class
              Collider.getColliderForPlayer(...)\n");
287     }
288
289     return *collider;
290 };
291
292 SDL_Rect Collider::getColliderForPlayerSword(int state, int curr_sprite,
      bool headLeft, SDL_Rect appearance) {
293     SDL_Rect* collider = NULL;
294
295     if ((state == Player::STAB) || (state == Player::THROWBOTTLE)) {
296         if (headLeft == true)
297         {
298             collider = getColliderForPlayerSwordHeadLeft(curr_sprite,
                  &appearance);
299         }
300         else if (headLeft == false) {
301             collider = getColliderForPlayerSwordHeadRight(curr_sprite,
                  &appearance);
302         }
303
304         if (collider == NULL) {
305             printf("Failed to return Collider from class
                  Collider.getColliderForPlayer(...)\n");
306         }
307     }
308
309     if (collider == NULL) {
310         collider = new SDL_Rect();
311         collider->x = 0;
312         collider->y = 0;
313         collider->h = 0;
314         collider->w = 0;
315     }
316
317     return *collider;
318 };
```