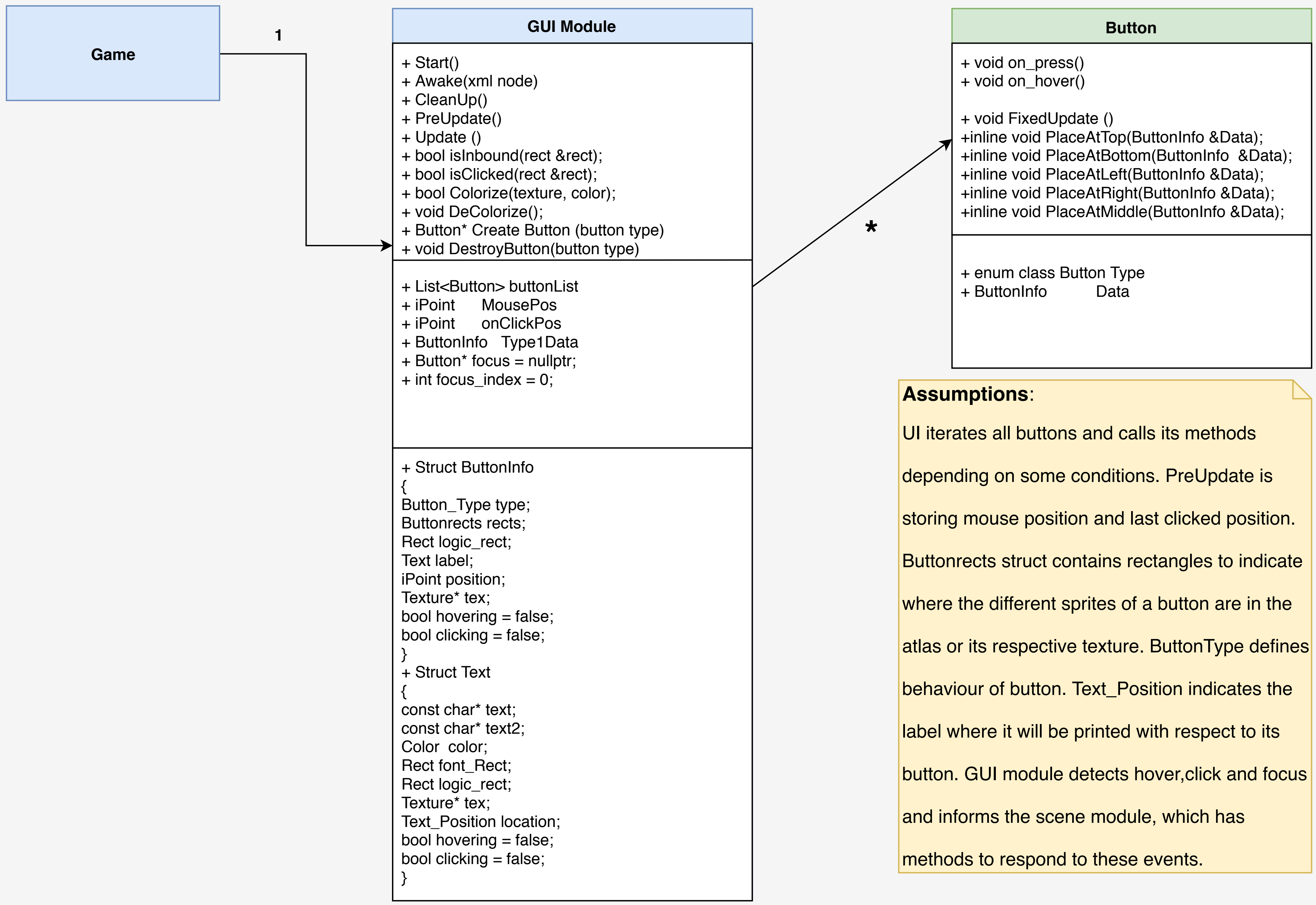
GUI UML



GUI Module

```
+ Start()
+ Awake(xml node)
+ CleanUp()
+ PreUpdate()
+ Update ()
+ bool isInbound(rect &rect);
+ bool isClicked(rect &rect);
+ bool Colorize(texture, color);
+ void DeColorize();
+ Button* Create Button (button type)
+ void DestroyButton(button type)
```

```
+ List<Button> buttonList
+ iPoint      MousePos
+ iPoint      onClickPos
+ ButtonInfo   Type1Data
+ Button* focus = nullptr;
+ int focus_index = 0;
```

```
+ Struct ButtonInfo
{
Button_Type type;
Buttonrects rects;
Rect logic_rect;
Text label;
iPoint position;
Texture* tex;
bool hovering = false;
bool clicking = false;
}
+ Struct Text
{
const char* text;
const char* text2;
Color  color;
Rect font_Rect;
Rect logic_rect;
Texture* tex;
Text_Position location;
bool hovering = false;
bool clicking = false;
}
```

Button

```
+ void on_press()
+ void on_hover()

+ void FixedUpdate ()
+inline void PlaceAtTop(ButtonInfo &Data);
+inline void PlaceAtBottom(ButtonInfo  &Data);
+inline void PlaceAtLeft(ButtonInfo &Data);
+inline void PlaceAtRight(ButtonInfo &Data);
+inline void PlaceAtMiddle(ButtonInfo &Data);
```

```
+ enum class Button Type
+ ButtonInfo          Data
```

Assumptions:

UI iterates all buttons and calls its methods depending on some conditions. PreUpdate is storing mouse position and last clicked position. Buttonrects struct contains rectangles to indicate where the different sprites of a button are in the atlas or its respective texture. ButtonType defines behaviour of button. Text_Position indicates the label where it will be printed with respect to its button. GUI module detects hover,click and focus and informs the scene module, which has methods to respond to these events.