MENU SYSTEMS
MENU STACK

• Ensure a menu system has a stack
  • Need base class for menus

• Allows going back to a prior menu selection
  • Its like allowing a user traversal of a tree

• On entering a new menu – push

• On exiting the current menu – pop

• Good to offer the ability to go to top menu
Each button needs at least two visual states:
- Selected
- Unselected
- Maybe pressed state as well

Possibly have a doubly-linked list of buttons or an array of buttons tracking an index:
- Back
- Forward
- Wrap

Possibly have a 2D bounding box for clicking or tapping

Event system to manage what happens when a button is pressed.
• Supports allowing someone to customize names for avatars
• You can only allow one key stroke at a time, so you construct a string as the player presses keys
HUD ELEMENTS
HUD ELEMENTS

• The HUD or Heads Up Display displays pertinent information to a player for each gameplay mode
• It is an overlay on top of the view of the game
• Simple elements
  • Buttons
  • Score
  • Health bars
WAYPOINT ARROW

- Store vector for facing direction $\hat{f}$
- Compute vector to target $\hat{t}$
- Angle of rotation
  - $\theta = \cos^{-1}(\hat{t} \cdot \hat{f})$
- Axis of rotation
  - $\hat{a} = \hat{t} \times \hat{f}$
- Rendering considerations
  - Should not be affected by camera transformation
  - Should not be affected by z-buffering
AIMING RETICULE

• Drawn as a crosshair at a set 2D position
• Ray cast is performed into the scene from the unprojected 2D position
• Depending on what the ray hits you change the color/shape of the reticule
RADAR

- Convert player and objects of interest into 2D positions
- Determine distance and vector to objects of interest
  - Draw blip if inside view based on target vector
OTHER CONSIDERATIONS

• Design in relative coordinates to support multiple resolutions
• Remember to support localization
• Use middleware for the UI as much as possible
• Design for user experience!
GPAT – CHAPTER 11 (NOT ASSIGNED IN READING)
SCRIPTING LANGUAGES AND DATA FORMAT
SCRIPTING LANGUAGES

• Allows designers to get involved in the programming
  • Abstract the engine (hard stuff) from the game elements ("easier" stuff)

• Use a scripting language that is interpreted/compiled by the engine
  • Allows easy updates to the game to be distributed
  • Can reload script dynamically for debugging
  • Prevents crashes
  • However, can be slow
IMPLEMENTING A SCRIPTING LANGUAGE

- Lexing
  - Tokenization (Lexical analysis) – make "tokens" out of a stream of text. Typically done through regular expressions.
    - Operators
    - Identifiers
    - Keywords
    - Etc

- Parsing
  - Syntax analysis – ensure tokens follow rules of the language. Typically done through context-free grammars.

- Executing
  - Code execution/generation
DATA FORMATS

• Binary file – unreadable file that stores values of the bits directly
  • Efficient
  • Needs some way to help debug/designers

• Text-based file – readable file that stores values of the bits as strings
  • Easy editing for designers and repositories
  • Allows end users to modify (user mods)
  • Can use standard options
    • XML
    • JSON

• Both – text-based in development and binary in release
DATA FORMATS

XML

```xml
<empinfo>
   <employees>
      <employee>
         <name>James Kirk</name>
         <age>40</age>
      </employee>
      <employee>
         <name>Jean-Luc Picard</name>
         <age>45</age>
      </employee>
      <employee>
         <name>Wesley Crusher</name>
         <age>27</age>
      </employee>
   </employees>
</empinfo>
```

JSON

```json
{  
   "empinfo": {
      "employees": [
         {
            "name": "James Kirk",
            "age": 40,
         },
         {
            "name": "Jean-Luc Picard",
            "age": 45,
         },
         {
            "name": "Wesley Crusher",
            "age": 27,
         }
      ]
   }
}
```
WORK ON DESIGNING YOUR MENU SYSTEM AND HUD FOR YOUR PRIMARY GAMEPLAY MODES
In this chapter, we looked at some basic approaches to defining and implementing a user interface for a game.

- Menu systems
- HUD elements