Immediate Mode GUI – Theory and Example

Adam Sawicki

http://asawicki.info
What is GUI?

Graphical User Interface (GUI)
aka Head-Up Display (HUD)

• Displays information
• Handles interaction
  • Mouse, touch, keyboard, other controllers...
Implementation

1. Default system controls
   • (Used in desktop apps)
   • Pure system API: WinAPI
   • C++ library: Qt, wxWidgets, ...
   • Another programming language: C#, Java, ...

[Image of search settings]
Implementation

1. Default system controls
2. Custom rendering
   • (Used in games, graphics apps)
   • Library: Scaleform, Dear ImGui, ...
   • Your own implementation
Architecture

Object-oriented – seems to be the most natural architecture for GUI

- Class hierarchy for types of controls
- Design patterns, e.g. composite
- Fields: Position, Size, ...
- Methods: Show, Hide, Enable, Disable, ...
Architecture

Object-oriented – seems to be the most natural architecture for GUI
Example

Oto wpisany tekst Hello World!
Some thoughts

• Actual objects also form a hierarchy
  • Every control is positioned relative to its parent

• Similar to how games represent scene
  • High-level approach found in game engines – hierarchy of objects
Immediate mode GUI – idea

- On low level (DirectX, OpenGL, Vulkan), rendering is stateless – „immediate mode”
  - Sequence of draw calls repeated every frame

```plaintext
- SetShader
- SetTexture
- DrawTriangles
- SetTexture
- DrawTriangles
- ...
```

- What if... we could render GUI this way?
Dear ImGui

- [https://github.com/ocornut/imgui](https://github.com/ocornut/imgui)
- C++ library
  - Bindings to many languages available
- License: MIT
- Author: Omar Cornut (game developer)
- Suited for real-time rendering
  - Efficient
  - Graphics API agnostic
Example code

```cpp
if(ImGui::Begin("First Window"))
{
    ImGui::Text("Do you like it?");
    ImGui::Button("Yes");
    ImGui::Button("No");
    ImGui::End();
}
```
Q: How are controls positioned?

• Automatic – each control in new row
• You can change this:
  • PushItemWidth(item_width), SameLine()
  • Columns: Columns(count)
  • Grouping: BeginGroup(), EndGroup()
  • Full control: GetCursorPos(), SetCursorPos(local_pos)

Lists:
<table>
<thead>
<tr>
<th>AAAA</th>
<th>AAAA</th>
<th>AAAA</th>
<th>AAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBBB</td>
<td>BBBB</td>
<td>BBBB</td>
<td>BBBB</td>
</tr>
<tr>
<td>CCCC</td>
<td>CCCC</td>
<td>CCCC</td>
<td>CCCC</td>
</tr>
<tr>
<td>DDDD</td>
<td>DDDD</td>
<td>DDDD</td>
<td>DDDD</td>
</tr>
</tbody>
</table>
Q: How do I handle feedback?

You receive it the same moment you define a control

ImGui::Text("Do you really want to quit?");
if(ImGui::Button("Yes"))
    ExitProgram();
if(ImGui::Button("No"))
    CloseThisWindow();
Q: Where is value of controls?

• In your own variables
• You pass pointers to them as you define controls

```cpp
float volume = 0.7f;
bool mute = true;

ImGui::SliderFloat("Volume", &volume, 0.0f, 1.0f);
ImGui::Checkbox("Mute", &mute);
```
Q: How about other state?

- There is other state
  - Window position & size, focus, text selection, ...

- Kept inside ImGui library
  - So it’s not truly stateless...

- Controls are identified by hash from their labels
  - If not unique, you can use "Label##UNIQUE_ID"
  - You can also scope your labels: PushID(), PopID()
Q: How to render?

• If your frame has Update() and Render():
  • Do all ImGui inside Update()
  • Inside Render() you must render it yourself

• For rendering you receive a sequence of:
  • Vertex array
  • Index array
  • Texture ID
  • Scissor rectangle

• Examples available for: DirectX 9, 10, 11, OpenGL, Vulkan, Allegro, ...
Features: Property grid

• Suited to provide editing of properties of various types
  • bool, int, float, string, enum
  • vec2, vec3, vec4, color
Features: Fonts

• Supports TTF fonts, loaded into atlas texture
• You receive data for this texture, need to upload it to GPU by yourself
Live demo
Conclusion

• Dear ImGui is good for:
  • Internal in-game GUI for debugging purposes
    • Easy to use
    • Efficient to render

• Dear ImGui is not good for:
  • Fully-featured game editor
    • Not well suited for complicated GUI, better use system controls
  • Final game HUD
    • No possibility of skinning or making it look pretty

• You can always make your own implementation
  • Based on idea of Immediate Mode GUI!
Thank you

Questions?