

Immediate Mode Graphical User Interfaces

Presented by

Rickard Gustafsson

Johannes Algelind

Traditional method

- Retained Mode Graphical User Interfaces
 - Uses Model View Controller
 - All your data lives in the GUI library
 - You must copy your data back and forth between application and library (position, size, text, ...)
 - Communicate with your widgets using ID's or objects
 - Big library with lot's of classes and code.

Why do we want a new way of doing GUIs?

- Retained mode is a pain
 - Message Queue / Callbacks
 - ID's / Objects
 - Decentralized linkage
 - Synchronization between GUI library and application (inherently because of MVC)
 - Data-driven
 - Adding custom widgets is tricky

Typical RMGUI usage example

```
WidgetSet* ws;
const unsigned int ID_BUTTON = 1;
const unsigned int ID_SLIDER = 2;
float my_float = 0.5f;

void init()
{
    ws = new WidgetSet();
    ws->add(create_button(ID_BUTTON, "Click me", ...));
    ws->add(create_slider_float(ID_SLIDER, "Slide me", 0.f, 1.f, ...));
    ws->set_float_value(ID_SLIDER, my_float);
    ws->set_callback(&callback_ws);
}

void callback_ws(int widget_id)
{
    switch(widget_id)
    {
        case ID_BUTTON: do_action(); break;
        case ID_SLIDER: my_float = ws->get_float_value(ID_SLIDER); break;
    }
}

void main()
{
    init();
    while(running)
        ws->draw();
}
```

Immediate Mode GUI

- Casey Muratori stumbled upon ImGui around 2002 while working at RAD Game Tools.
- A style of doing GUIs which are well suited for real-time rendered applications.
- Zero Memory Widgets – ImGui library for regular applications, appeared around 2003.

IMGUI usage example

```
void draw_gui(struct GUIState_t* state, float* my_float)
{
    if (do_button(state, "Click me", ...))
        do_action();
    do_slider_float(state, "Slide me", my_float, 0.f, 1.f, ...);
}

void main()
{
    GUIState_t* state;
    float my_float = 0.5f;
    while(running)
    {
        update(state, ...);
        draw_gui(state, &my_float);
    }
}
```

IMGUI traits

- No data synchronization needed, all your data is stored in the application.
- Stores very little state in the library
- Uses procedural function-calls as "widgets".
- Code-driven
- Centralized flow control
- Easy to dive in to, easy to extend.
- Must be redrawn every frame

Simple IMGUI implementation

- Basic concepts
 - A widget is considered "**hot**" if there is a possibility that it will be interacted with.
 - An "**active**" widget is currently engaged by the user.
 - Widget ID's are used for the library to keep track of hot and active widgets.

Simple IMGUI implementation contd.

```
struct GUIState_t
{
    void* hot_item;
    void* active_item;
    int mouse_x, mouse_y;
    unsigned mouse_buttons;
};

struct Rect_t
{
    float x, y, w, h;
};
```

```
bool is_button_down(GUIState_t* state, unsigned button)
{
    return (state->mouse_buttons >> button-1) & 1;
}
```

```
bool mouse_inside_rect(GUIState_t* state, Rect_t* rect)
{ /* point-in-rect-test */ }
```

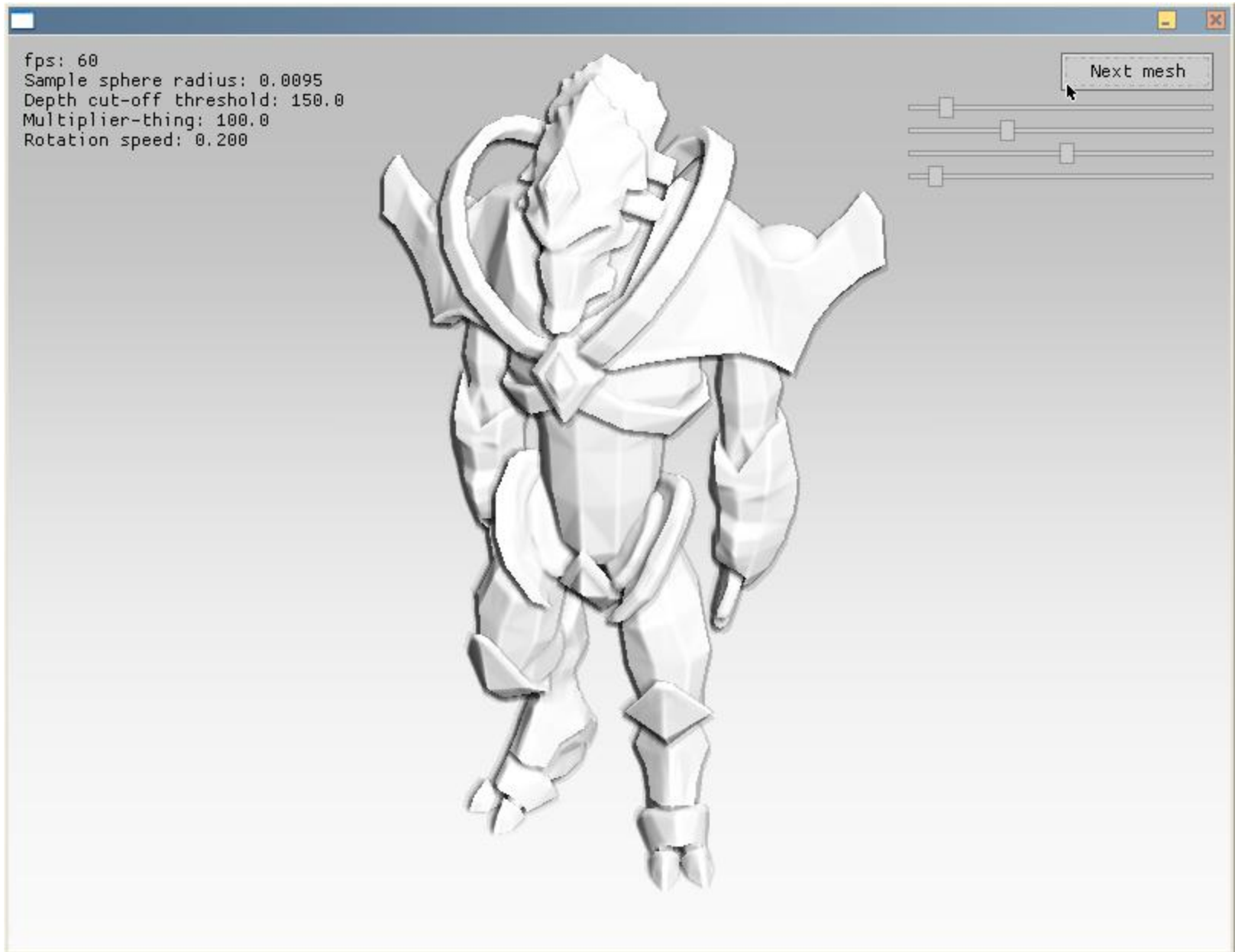
```
void update(GUIState_t* state, unsigned mb,
            int mx, int my)
{
    state->mouse_x = mx;
    state->mouse_y = my;
    state->mouse_buttons = mb;
}
```

```
bool do_button(GUIState_t* state, void* id, Rect_t* rect, const char* text, ...)
{
    bool result = false;
    bool inside = mouse_inside_rect(state, rect);

    if (inside)
        state->hot_item = id;

    if(state->active_item == id && !is_button_down(1))
    {
        if (inside)
        {
            result = true;
            state->hot_item = id;
        }
        else
            state->hot_item = 0;
            state->active_item = 0;
    }
    else if (state->hot_item == id)
    {
        if (is_button_down(1))
        {
            state->focused_item = id;
            state->active_item = id;
        }
    }
    draw_button(rect, text, ...);
    return result;
}
```

Demo-time



ImGui in Teeworlds

The screenshot displays the Teeworlds server browser interface. At the top, there are tabs for 'Internet', 'LAN', 'Favorites', 'Demos', 'Settings', and 'Quit'. The main area is a table of servers with columns for Name, Type, Map, Players, and Ping. A yellow mouse cursor is pointing at the 'set sail for fail CTF' server. To the right of the table is a 'Filter' panel with checkboxes for 'Has people playing', 'Server not full', 'No password', 'Compatible Version', 'Standard gametype', and 'Standard map'. Below the filter is a 'Game types' section with a '999 Maximum ping' button. At the bottom, there is a 'Quick search' field containing 'ctf2', a status bar showing '28 of 679 servers, 36 players', a 'Host address' field with '217.172.180.79:8303', and buttons for 'Refresh' and 'Connect'. The current version is '0.5.1'.

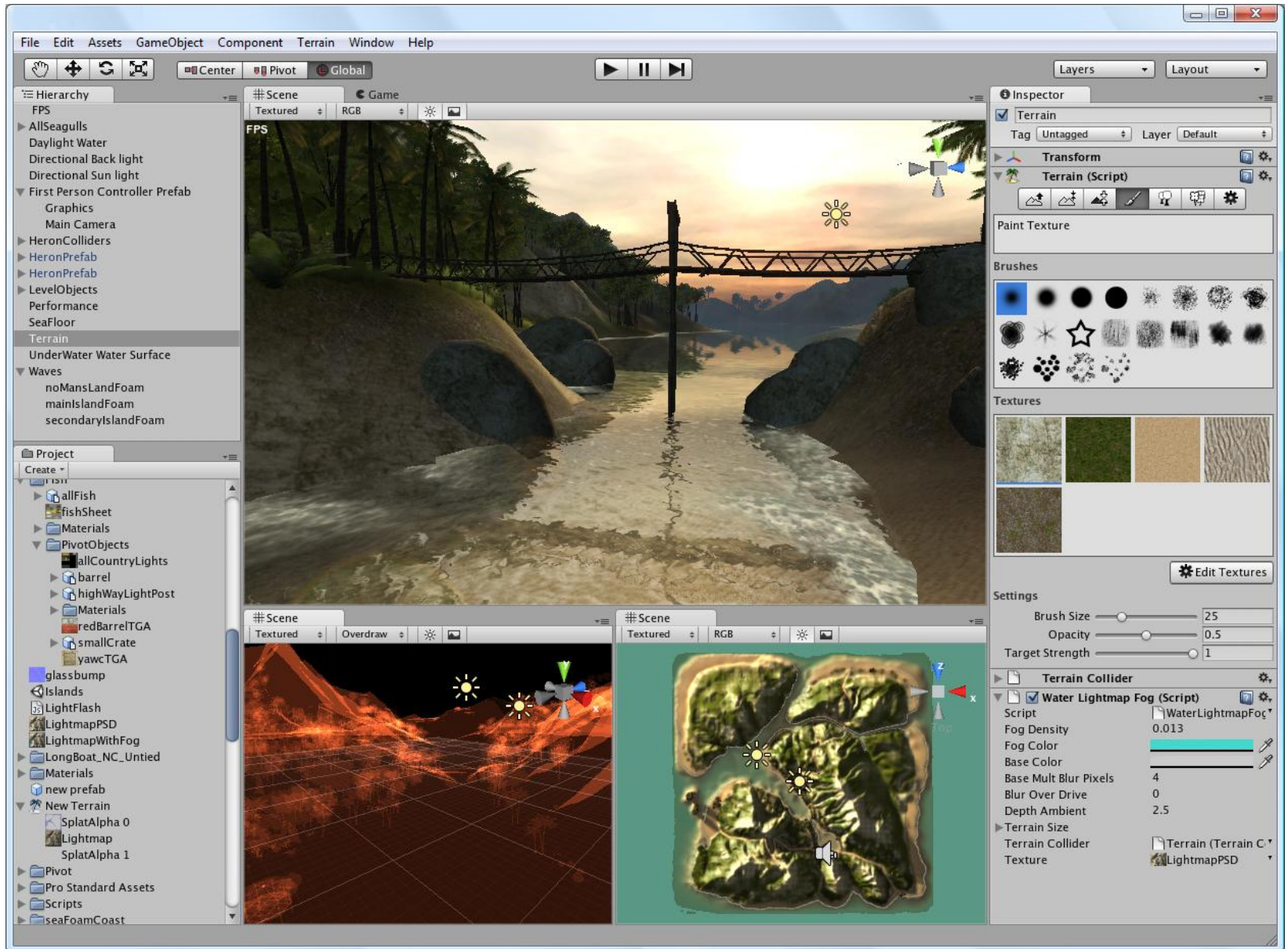
| Name | Type | Map | Players | Ping |
|---|------|------|---------|------|
| (SWE) CTF | CTF | ctf2 | 3/16 | 21 |
| *gV* - ctf ctf2 | CTF | ctf2 | 16/16 | 41 |
| TinyBox.net | CTF | ctf2 | 4/16 | 44 |
| [BiZzaro]ictf | DM | ctf2 | 0/16 | 49 |
| CTF Server by GameMods.org | CTF | ctf2 | 0/10 | 50 |
| DT-Elite | CTF | ctf2 | 0/16 | 52 |
| TieBones TeeWorld | DM | ctf2 | 2/16 | 52 |
| KirovNet.CTF Teeworlds Server | CTF | ctf2 | 0/12 | 56 |
| -CyberPain Teeworld CTF Server- | CTF | ctf2 | 0/16 | 65 |
| e66_tw_server | DM | ctf2 | 2/16 | 69 |
| GameZoo.it CTF | CTF | ctf2 | 0/12 | 77 |
| set sail for fail CTF | DM | ctf2 | 0/16 | 82 |
| MegaStyle TW CTF | CTF | ctf2 | 1/12 | 84 |
| Gagarino.NET CTF | CTF | ctf2 | 0/16 | 86 |
| Phönix CTF2 // by ZockerStube.net | CTF | ctf2 | 0/16 | 88 |
| bloodarmy.ru TW CTF | CTF | ctf2 | 0/16 | 96 |
| Games-by.net Teeworlds sample ctf | CTF | ctf2 | 0/8 | 113 |
| teeworlds.raspi.fi | CTF | ctf2 | 0/16 | 124 |
| GG.ST HS ctf server | CTF | ctf2 | 0/16 | 130 |
| *gV* - cKz Clanserver | CTF | ctf2 | 6/16 | 130 |
| CoCoS [RUS] Teeworlds CTF | CTF | ctf2 | 0/8 | 134 |
| ROSTOV dm | DM | ctf2 | 0/12 | 135 |
| srsgames.net teamplay vanilla CTF | CTF | ctf2 | 0/16 | 142 |
| 6on6 BattleTees #1 (Qi Clanserver) | CTF | ctf2 | 2/16 | 145 |
| [Qi] High level only (pass= qiRox) | CTF | ctf2 | 0/16 | 148 |
| EFS-Teeworlds-Server | CTF | ctf2 | 0/12 | 162 |
| AB Public Server 24/7 ctf2 | CTF | ctf2 | 0/16 | 166 |

Quick search: 28 of 679 servers, 36 players

Current version: 0.5.1

Host address:

Unity



What's next?

- Partitioned ImGui ("PImGui")

Sources & more info

- "Immediate Mode GUIs" written by Sean Barret for [GDM](#) Sept. 2005.
- [Introduction video](#) by Casey Muratori
- The [Molly Rocket forums](#)
- [Book in progress](#) by Johannes Norneby
- [Tutorial](#) by Jari Komppa
- [IMGUI implementation by nVidia](#)
- [Zero Memory Widgets](#)

Questions?