The base Quake 2 UI system is inflexible, very code-heavy, has no way for the server to tell a client which UI to draw and impossible to modify from game code. Zombono uses a new UI system that is designed to rectify this issue – it’s based on the concept of **controls** and **UI scripts.** It is also designed to rectify the severe issues that Zombono-Q1’s UI system had, such as the server depending on variables only the client can possibly understand.

**Contents**

1. **UI Overall Design**

UI is handled and drawn on the client side, with events sent to the server for processing. All UI is handled in C code that calls the **UI\_** client functions (with **UI\_Init** being called to initialise the UI system). Each UI (consisting of a **ui\_t** structure) is created at runtime to reduce overhead related to creating UIs during gameplay. Each UI is made up of a series of **controls** (all implemented in the monolithic **ui\_control\_t** struct, in the array **controls** of a **ui\_t** structure; presently there are a maximum of 32 chara), and **event handlers** that attach to various events. When a UI is being displayed, the **current\_ui** global variable is set to that UI. It is set to **NULL** when a UI is not being displayed.

For a UI system to be useful, it must also use events. An **event** is an occurrence that allows a UI to do something when it happens, and it is attached to the UI system’s event loop (which is a part of the regular Zombono Engine event loop).

1. **UI Controls**

A UI control is an individual element of a UI. It must be contained within a **ui­\_t** element via its **controls** array; there are several types of controls that each use their own constituent parts of the larger **ui\_control\_t** structure (the reason this is a monolithic design is in order to ensure consistency with the rest of the engine – especially the entity and edict structure, which uses a monolithic design of this type – “either fit it in **edict\_t** or don’t do it”). A control is drawn as part of the larger **UI\_Draw** loop during the game’s drawloop – it is not drawn while a legacy menu is active to reduce conflicts with legacy menus (of which there are two separate systems, one for drawing the menus and one for drawing the in-game HUD).

* 1. **UI Controls: Text**

Draws text using **conchars** or, in future, the TTF font engine. Ignores size for the time being as all text is **(8 \* 8) \* hudscale** cvar value in size.

**Properties**:

**Text** – The text that will be displayed.

* 1. **UI Controls: Image**
  2. **UI Controls: Slider**
  3. **UI Controls: Checkbox**
  4. **UI Controls: Box**
  5. **UI Controls: Separator**

1. **UI Events**
2. **UI Scripts**
3. **UI Event Handling**
4. **UI Notes (e.g. Functions)**