COHERENT:LABS

Coherent GT Workflow Guide

Contents

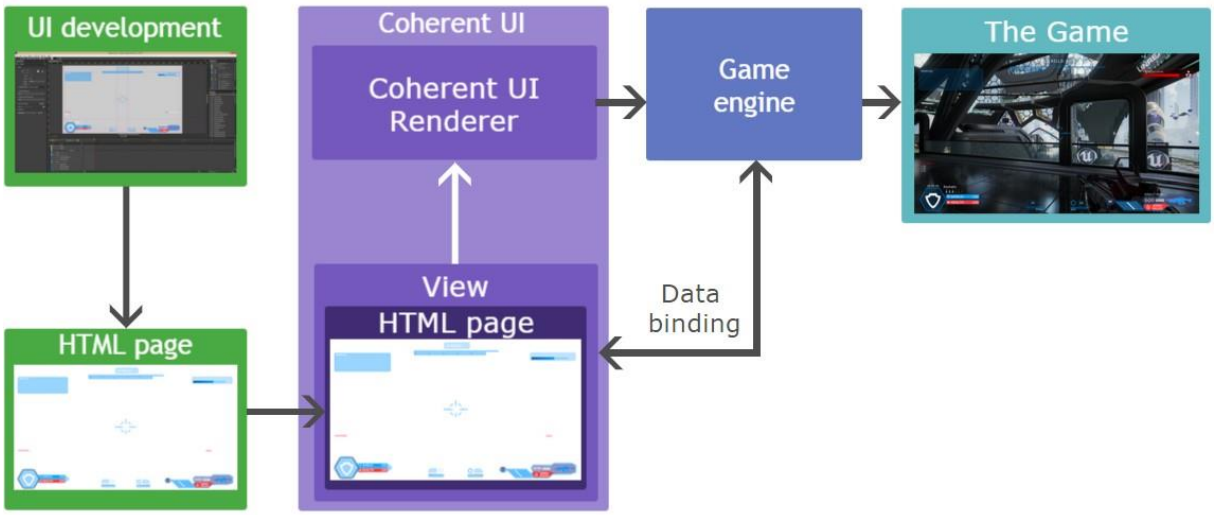
Chapter 1 – Workflow guide

1.1 About this document

The purpose of this document is to guide you through the UI development process. It goes over all the major phases and describes the software and technologies used. Please note that this a general workflow guide. For detailed documentation of Coherent GT please refer to the Coherent GT Documentation chm file or the documentation section on coherent-labs.com.

1.2 Introduction to Coherent GT

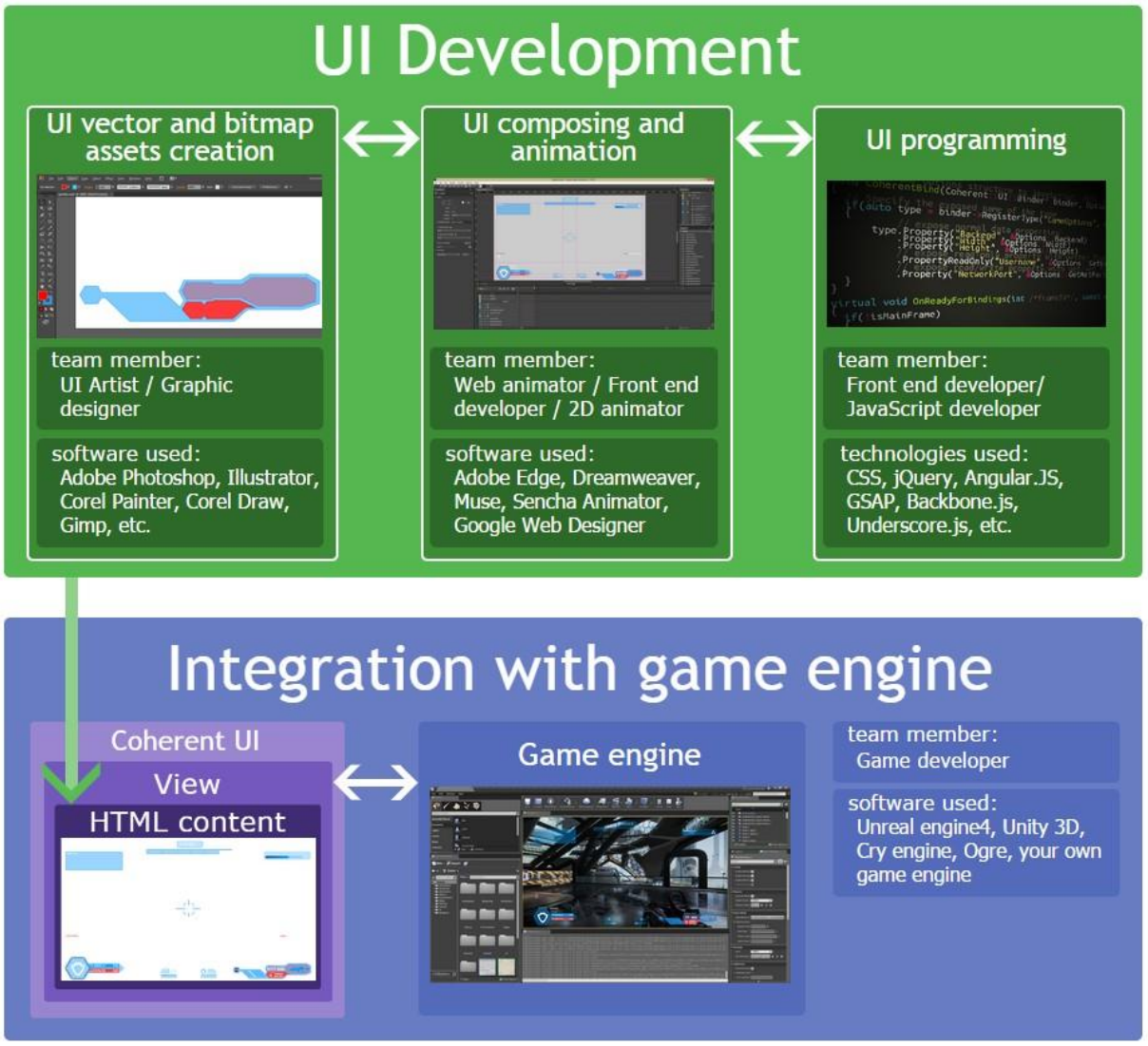
Coherent GT is a graphical user interface system specially designed for games. Game developers and UI artists can use standard modern HTML, CSS, and JavaScript to design and implement game UI and interaction.



Above you can see the basic functional diagram of Coherent GT. Essentially the result of the **UI development** (discussed further on in detail) is a HTML page that is placed inside of Coherent GT View. The Coherent GT **View** is basically a HTML5 page and the JavaScript context for it. The **View** allows you to perform various operations on the page, such as resizing, navigating to a different URL, sending input events, executing custom JavaScript code and so on. The HTML content in the **View** is rendered by the **Coherent GT Renderer** to a texture that is passed to the **game engine** which displays it in the **game**.

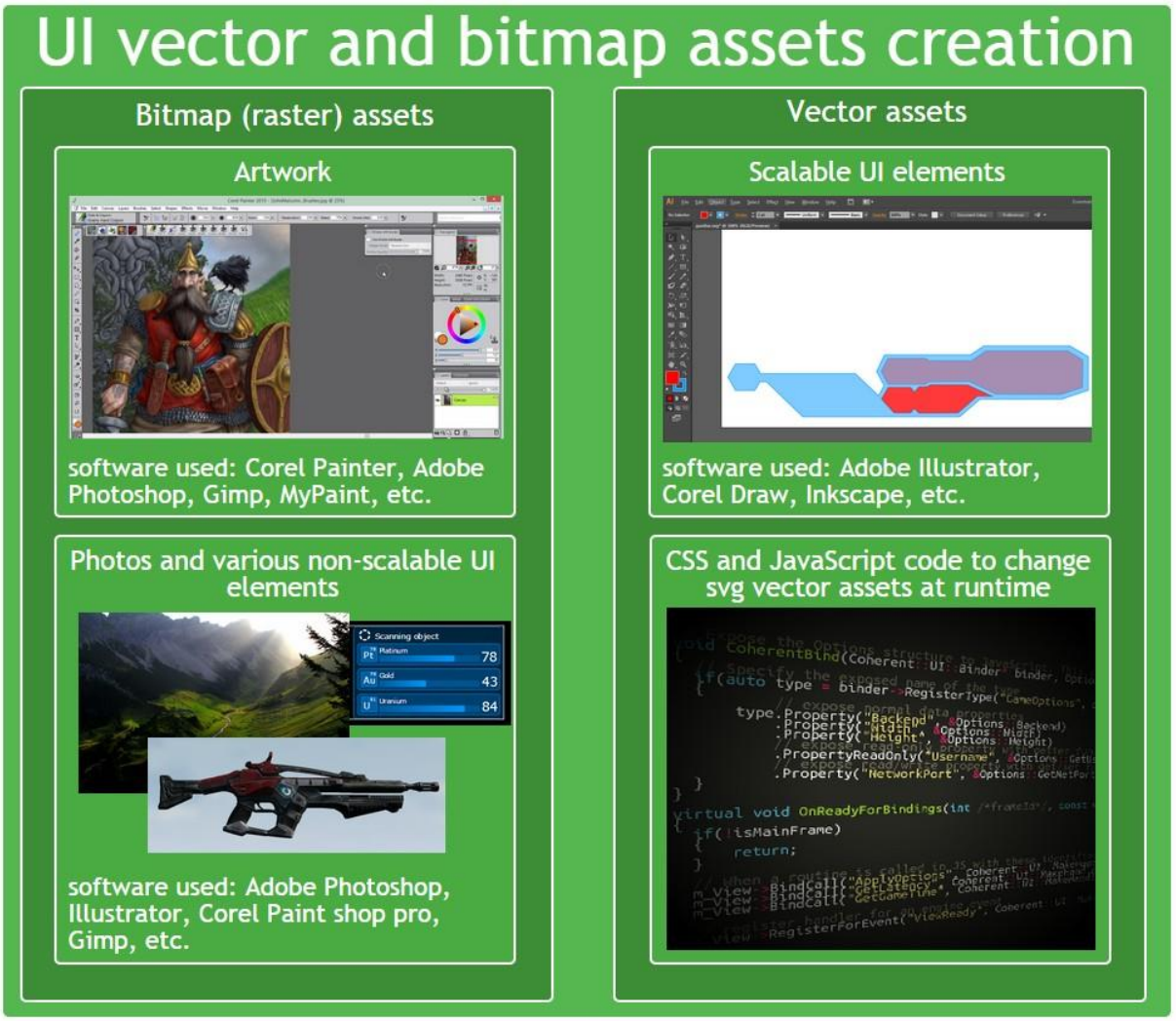
1.3 UI development

Major phases and tasks distribution in UI development.



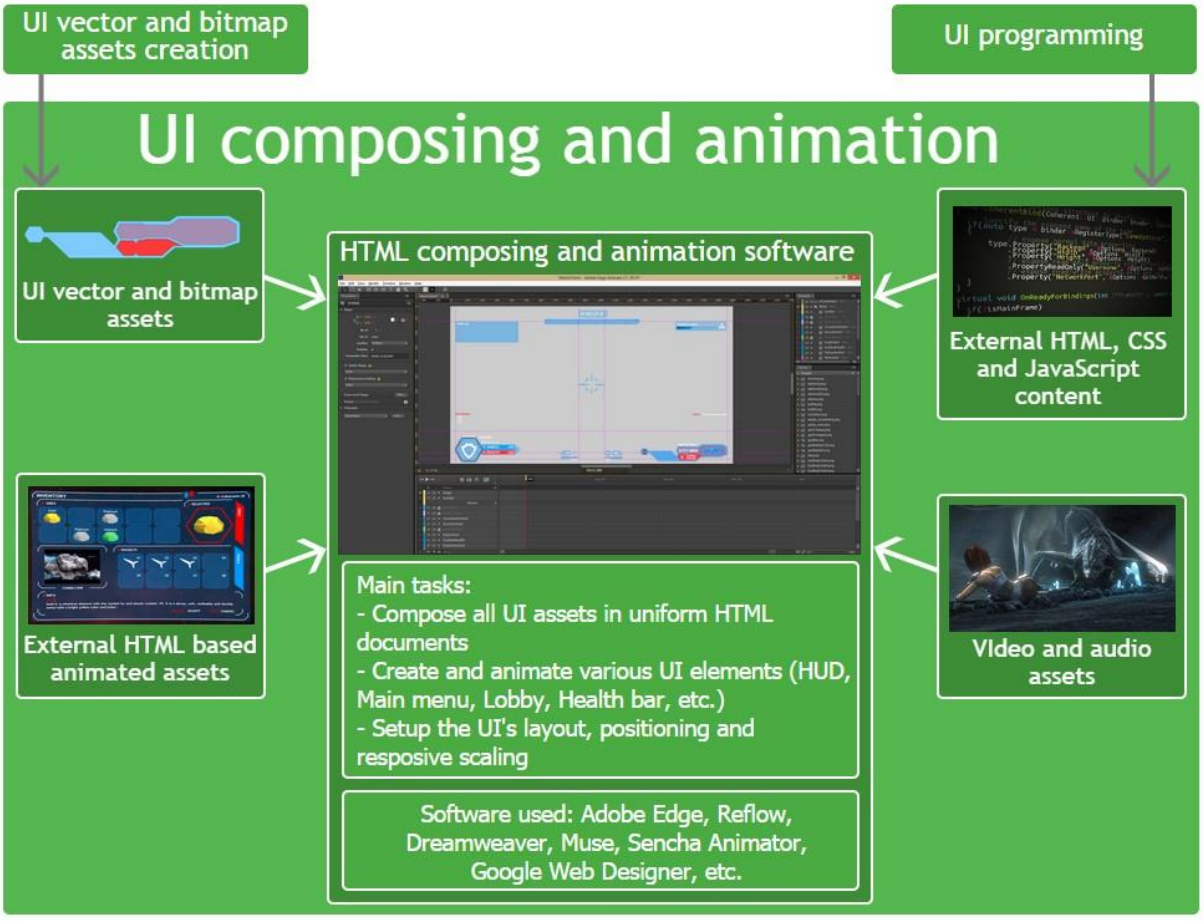
The HTML UI development process can be broken down in three major tasks – **UI vector and bitmap assets** design, **UI composing and animation** and **UI programming**. All three tasks can be carried out in parallel and the combined result is HTML page/pages that are displayed in **Coherent GT** view/views that are integrated in the game engine. The tasks are discussed in greater detail further on in the guide but in the diagram above you can see the team member involved and the software used for each phase.

UI vector and bitmap graphics assets creation



Although you can create basic shapes such as circles, rectangles, ellipses directly with HTML elements very often you'll need more complex shapes for your UI. These shapes are typically created as **bitmap** (raster) or **vector** images by UI artist or a graphic designer. **Bitmap** (raster) graphics images broadly speaking represent the image as rectangular grid of pixels. Bitmap images are used for all sorts of UI elements: artwork, photos and detailed UI elements. Their major drawback is that their quality degrades if they are scaled. **Vector** images on the other hand are based on geometrical primitives and scale perfectly regardless of the size of the UI. Furthermore the file size of the generated files is smaller than that of raster images. **Coherent GT** even supports **SVG** vector images that can be modified by CSS/JS at runtime (check the SVG images section for more information). The major drawback of vector images is that they are actually rendered at runtime and complex images might have effect on the performance. It is advisable to use vector images only when the UI has to be scaled for large resolutions or when you design simple shapes. On the diagram above you can see the software used for the different types of UI image assets.

UI composing and animation



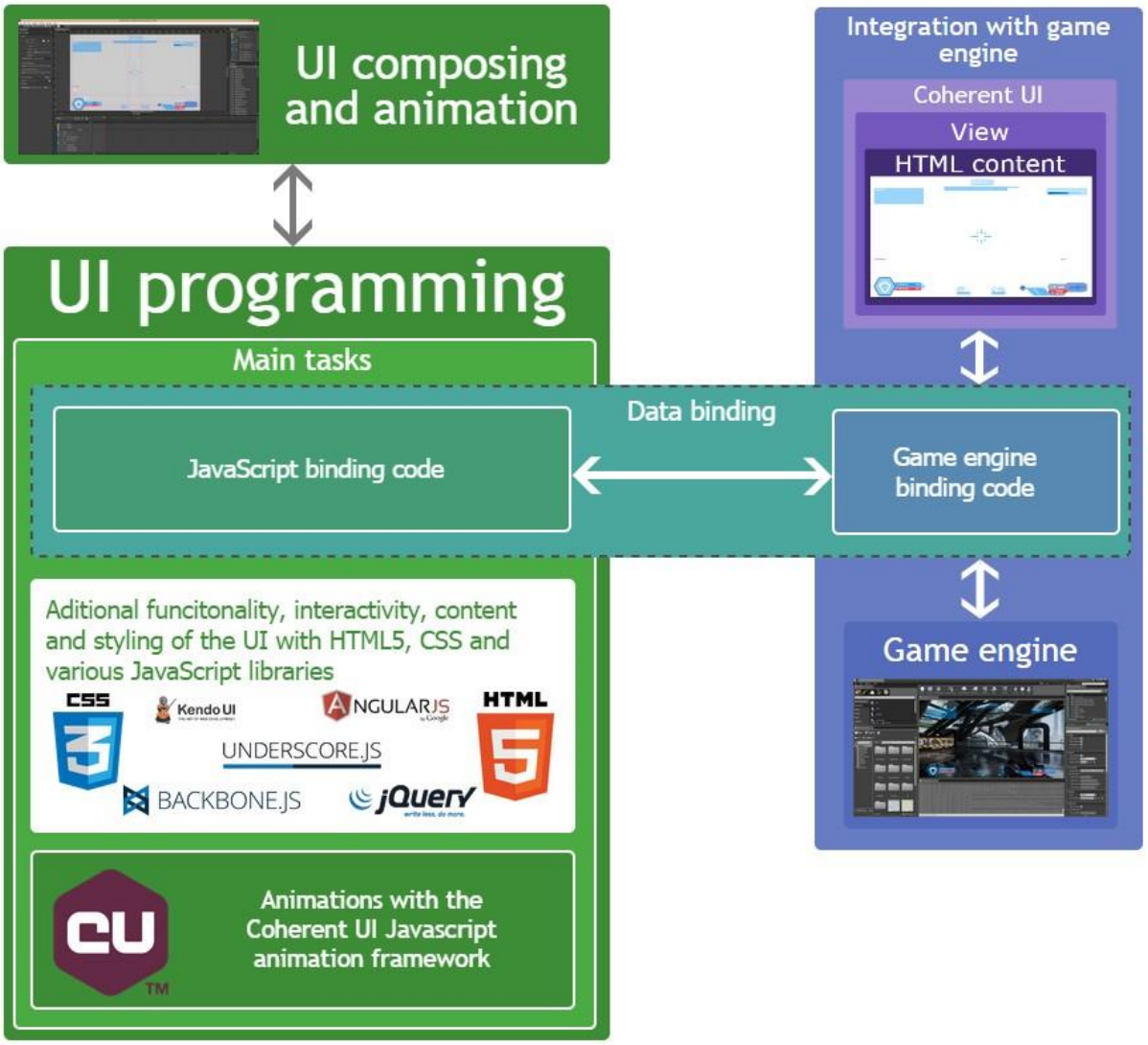
This stage is typically carried out by Web animator, Front end developer or 2D animator. It has three main tasks:

- **Compose all the UI assets** (vector and raster images, video, audio, external HTML/CSS/JS) in a uniform HTML documents

- **Create and animate various UI elements** (HUDs, Menus, Lobby, Healthbars, progress bars, etc)

- **Setup the UI's layout**, positioning and responsive scaling There are many **HTML** editors that you can use for this stage: Adobe Edge Animate, Edge Reflow, Dreamweaver, Muse, Sencha Animator, Google Web Designer, etc. However in our tests we've found that currently Adobe Edge has the largest set of useful features and it has excellent integration with the Adobe Creative Suite. If you like to know how to design your UI with Adobe Edge you can check these tutorials in the Coherent Lab's blog - tutorial 1, tutorial 2.

UI programming



This stage is very closely related to the UI composing and animation stage and it's typically carried out by a Front end developer or a JavaScript developer. It has three main tasks:

- Set up the **JavaScript** part of the code for the data binding
- Add additional functionality, interactivity, content and styling of the HTML pages using HTML5, CSS and various JavaScript libraries
- Add additional animations with the **Coherent GT JavaScript animation framework**

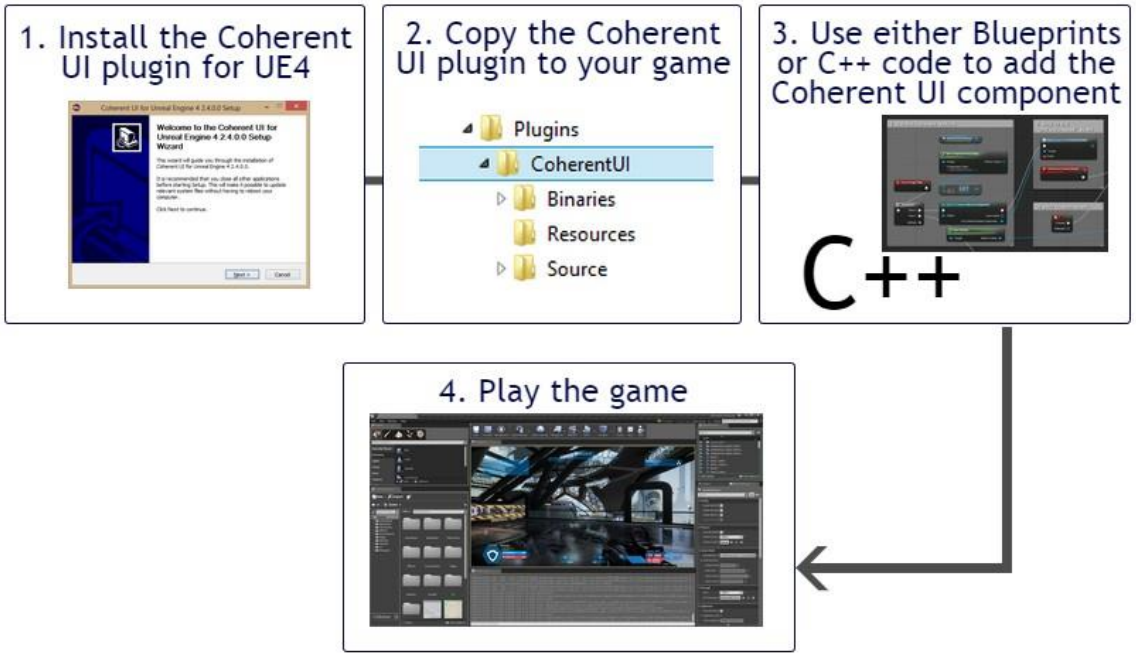
In this stage you can use the Coherent GT Debugger in collaboration with the game developer to debug for HTML/- CSS/JS errors.

1.4 Integration with game engine

This phase is typically carried out by a game developer. He is responsible for adding the Coherent GT **View** component and setting its settings in the game engine. Furthermore the game developer also writes

the game engine binding code. Coherent GT is pre-integrated with **Unreal engine 4** and **Unity 3D** but you can integrate it with other game engines as well.

Unreal Engine 4



To integrate Coherent GT in your Unreal engine 4 project first install the plug-in via the installer. Then copy the plug-in files to your project and use either blueprints or C++ code to add the Coherent GT **View component**. For more information please refer to the **Coherent GT UE4 guide**.

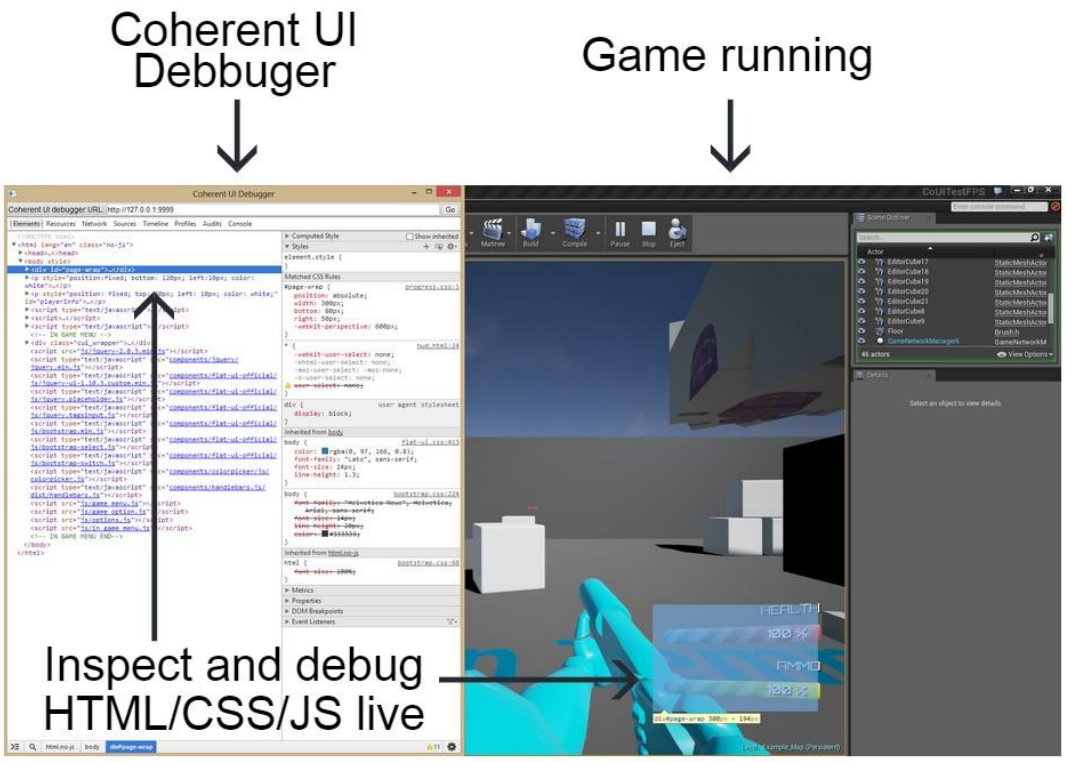
Other

Apart from Unreal engine 4 and Unity 3D you can easily integrate Coherent GT in your own game engine. Please follow the Coherent GT Quick Start Guide.

1.5 Additional UI Devleopment tips

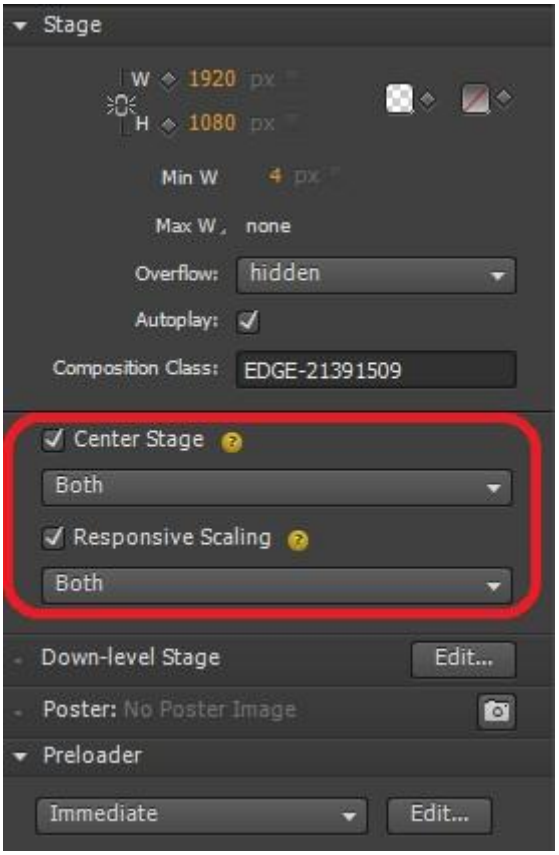
Live debugging

You can use the Coherent GT Debugger to inspect and debug your UI live. To do this just run your game (either in editor or as standalone), start the Coherent GT Debugger and you'll see a list of the HMTL pages of your UI. Choose one and you can track layout and styling issues, check for errors and set breakpoints for JavaScript, monitor loading time and many more and see the effect live. The Coherent GT Debbuger`s functionality is quite similar to the Dev Tool of Google Chrom.



Responsive scaling

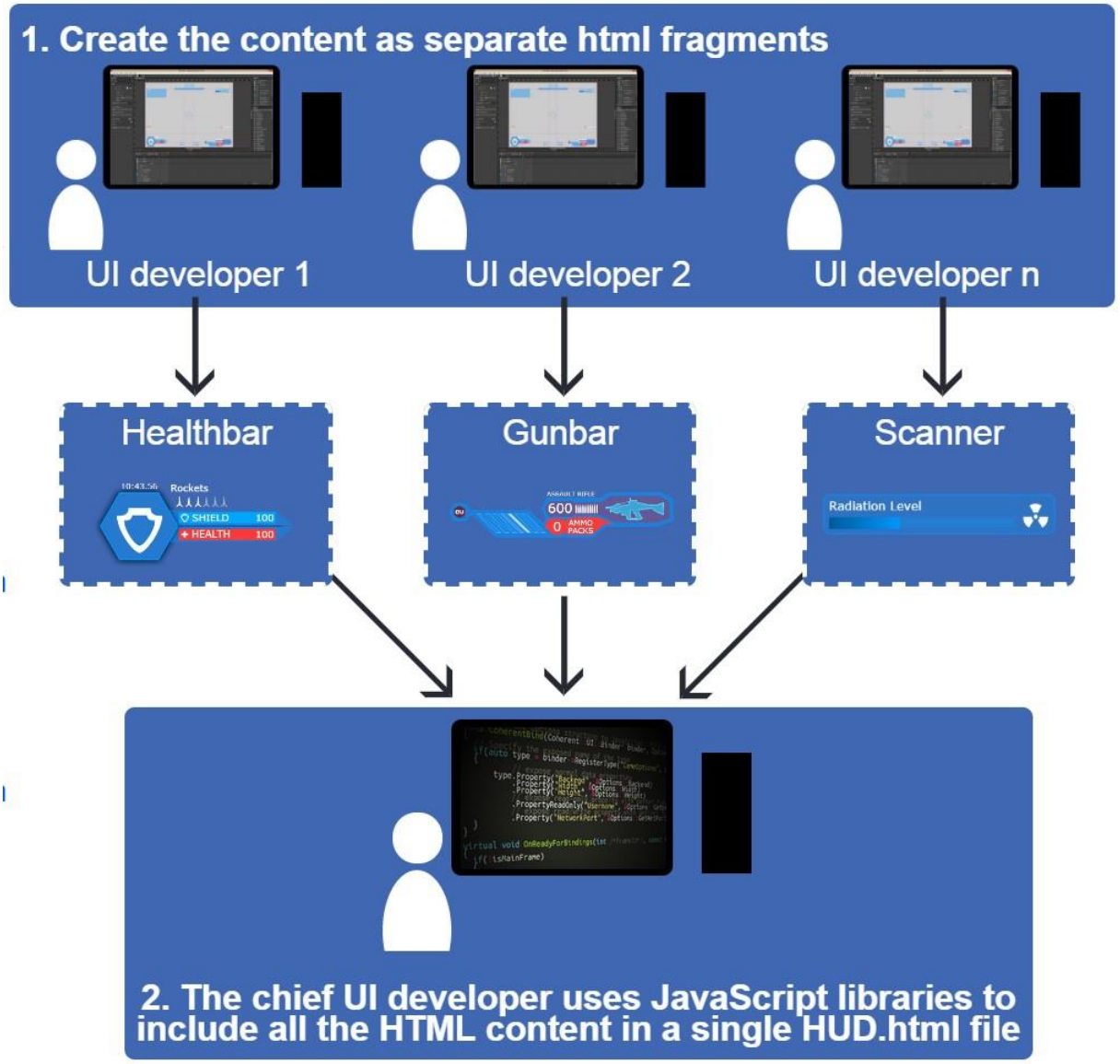
Adding responsive scaling functionality to your UI is essential to make it render properly regardless of the resolution. If you are composing your UI in Adobe Edge you can very easily do this.



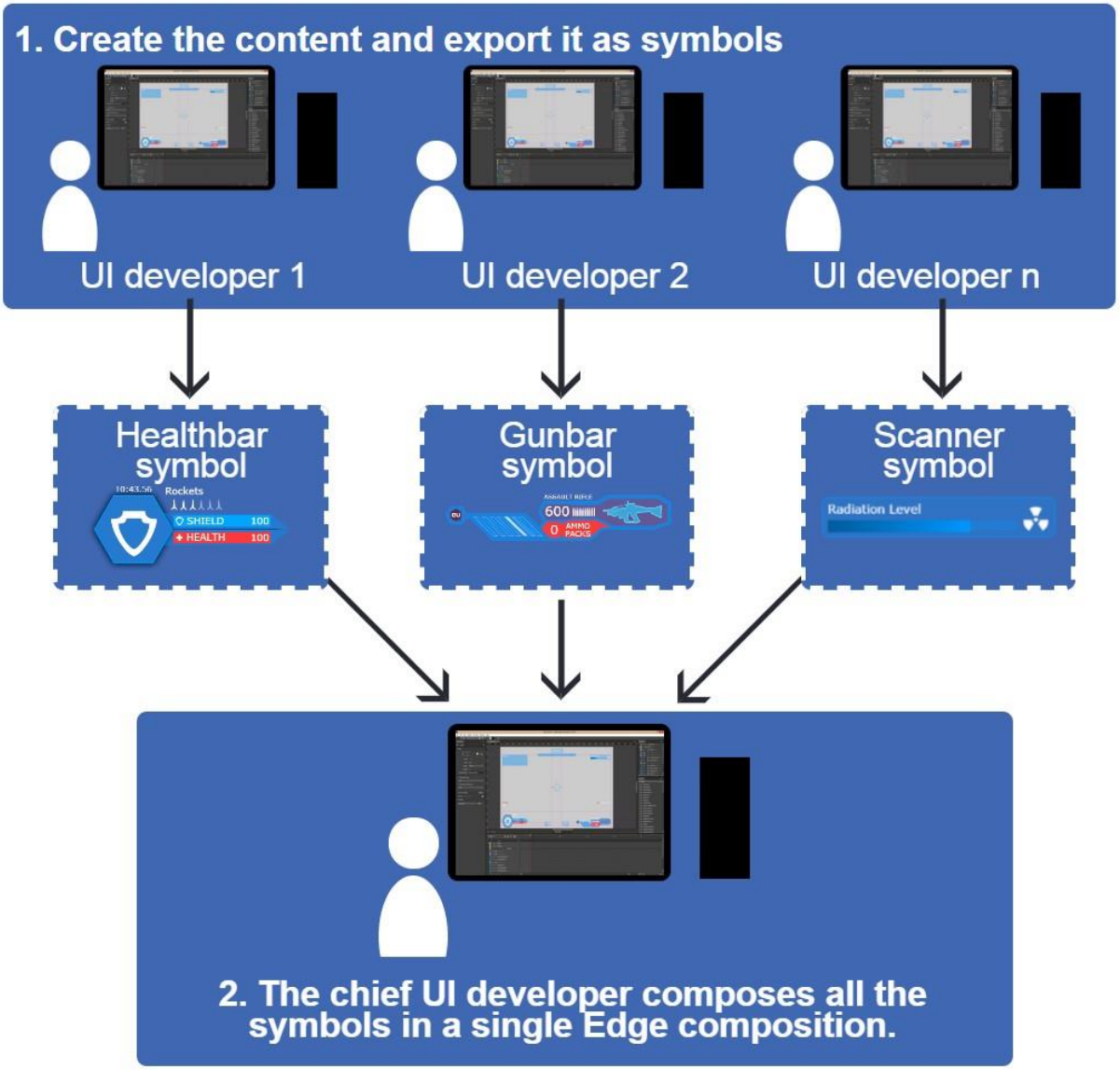
In the stage settings just check the “Responsive Scaling” and “Center Stage” checkboxes. That way your UI will be perfectly scaled and centered regardless of the resolution. Of course this is just one of the many possible solutions for scaling your HTML content. If you prefer not to use Adobe Edge you can setup the scaling directly in your HTML/CSS code. There are many ways to achieve this, such as using percentage values for width and height of the elements, CSS transform scale, CSS vw/vh units, “viewport” meta tags, etc.

Co-working on the same HTML document

Sometimes (especially in larger projects) you might have several UI developers working on the same HTML file. For example you might have a HUD.html file for a shooter game and you’ll want one UI developer to work on the minimap, one of healthbar and so on. To facilitate the process you can break the HUD page in several separate pages for each of the main elements (for example minimap.html, healthbar.html). Then just use JavaScript libraries like curl.js, require.js, or jQuery load to load them in the main HUD.html page.



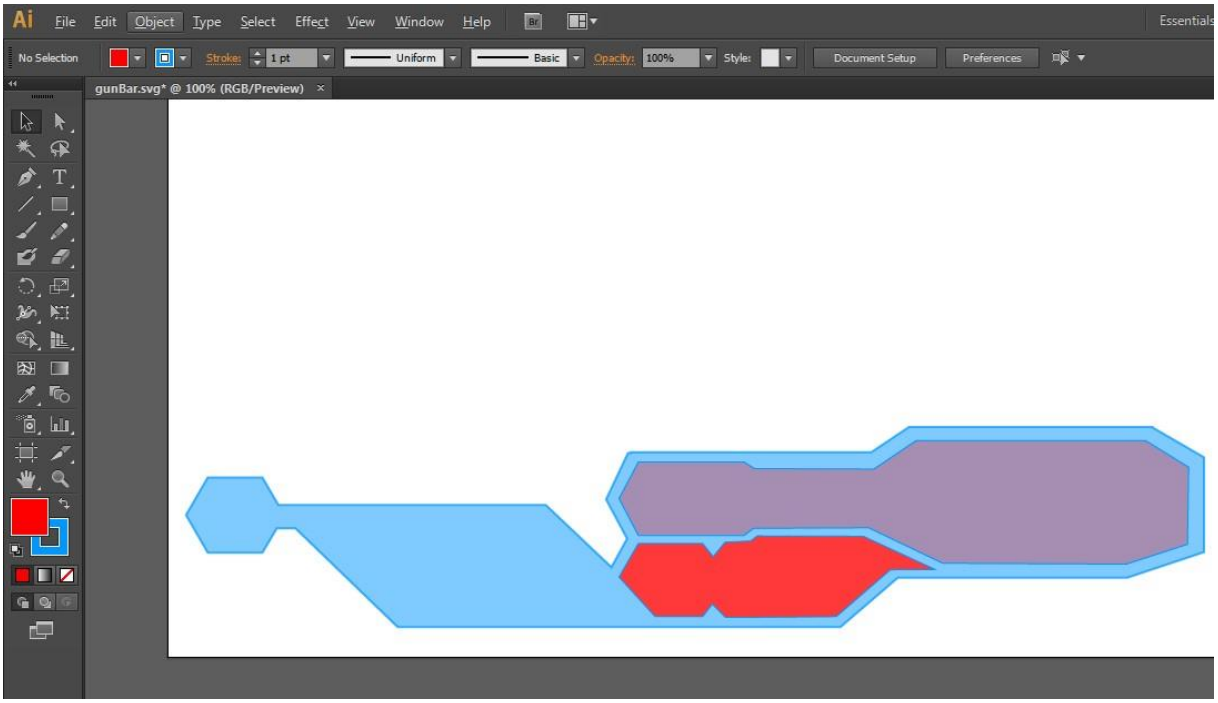
If you are using Adobe Edge each of the UI developers can export their part of the UI as symbols. Then the chief UI developer can compose all the symbols in a single Edge composition.



Local web server for faster UI development

During the development stage, you can host the currently developed HTML pages so that they can be accessed by other computers in your company. You can use simple web server like node.js and start it on the computer of the person developing the HTML page. Then just set the address of the Coherent View to match the IP address of the hosted page. That way the game developer can see the changes in the UI instantaneously. This can really speed up the workflow between the web designer and the game developer. After the UI is finalized just update the URL of the views to the local html files.

SVG images



Using **SVG** image assets has many advantages. You can directly copy/paste svg elements between Adobe Illustra- tor and Adobe Edge speeding up the design process. Adobe Illustrator has an"preserve illustrator editing" option when saving as svg. It allows you to make the exported svg files editable and use just one file (instead of for example editing .ai file and exporting as .png). Furthermore the size of the svg image files is very small compared to raster images. Being vector based svg images scale perfectly with no decrease in quality even at very large resolutions (4K and above). However note that complex SVG graphics can be computationally expensive so use them with caution.

You can even use **CSS/JS** to modify svg images at runtime. Here are a few excellent tutorials how to modify svg imags with CSS/JS:

- http://css-tricks.com/using-svg/
- http://apike.ca/prog_svg_js_create.html
- http://www.informit.com/articles/article.aspx?p=1609153

Improving your JavaScript with Google Closure compiler

Google Closure compiler (https://developers.google.com/closure/compiler/) is a tool for making JavaScript download and run faster. You can use it to reduce the size of your JavaScript, check for errors and improve the overall performance of your UI.

It has three optmizitaion levels:

- The WHITESPACE_ONLY compilation level – removes comments from yor code and also removes line breaks unnecessary spaces, extraneous punctuation (such as paranthesses and semicolons), and other whitespace
- The SIMPLE_OPTIMIZATIONS compilation level - performs the same whitespace and comment removal as WHITESPACE_ONLY, but it also performs optimizations within expressions and functions, including renaming local variables and function parameters to shorter names. Renaming variables to shorter names makes code significantly smaller. Because the SIMPLE_OPTIMIZATIONS level renames only symbols that are local to functions, it does not interfere with the interaction between the compiled JavaScript and other JavaScript
- The ADVANCED_OPTIMIZATIONS compilation level – performs the same transformation as SIMPLE_OPTIMIZATIONS, but adds a variety of more aggressive global transformations to achieve the

highest compressions of all thee levels. The ADVANCED_OPTIMIZATION level compresses JavaScript well beyond what is possible with other tools. Make sure to check the error tab as it can reveal all sorts of programming errors that are not parse error such as: using undeclared variable, unreachable code, reassigning constants, etc. To improve your code you can also code quality tools such as Google Closure Linter, isLint and jhHint