Presentable Portfolio Pieces

Table of Contents

In this session we’re going to cover some key areas of creating presentable portfolio pieces which a potential job may want to look at before hiring a candidate. Some companies if given the opportunity will want to look though a project you’ve already created and ask you questions about it either in interview, or even worse, look through the projects when comparing you to another candidate after this stage.

Engaging Portfolio Pieces

Adding fun to your portfolio pages — A couple extra points

- Max Amaden
  - Look at how animated and engaging these GIFs are for his games. Doesn’t this make you want to see more about this project?
- Unity WebGL Builds
  - If you have a simple quick to play unity project why not spend some extra time time to put a playable demo of your project on your website.
- Terry’s Portfolio page on Moths
  - Showing personality is good.
  - You can have Blogs material on anything that is slightly relevant to your career. Showing passion is always good.
- DannyDaley’s Project navigator

This site has the most professional projects front loaded but hit the second page and more projects that act as blogs become present.

- Creating engagement through stuff other than your projects such as articles can make you seem more professional and engaged in the field above most of competitors. Blog sections can help your website feel more populated (Though it helps to create seperate sections)
- This guy even had a store at one point same as nodes.
- If you use any platforms to show projects like github or artstation link to it.

Task

Plan some Portfolio Projects or Blog Idea's to help populate your site.

Clean GitHub Code

Many programming positions will want you to include a link to your github account. There is a chance that recruiters will look at the quality of your code on github before going forwards with interviewing you (Though not always). Though this doesn't always happen it helps to make sure that your most recent github projects reflect well of you.

Knowing how your code works is probably an obvious thing to remember before being questioned about it. But do you know how to explain it? Do you know how you would improve the feature or the way it was programmed if you had to implement it again. This is where code architecture can come in handy. Using linguistic tools such as common **design patterns** and highly readable code can be important for this. If not for the companies reference for your own if you're ever questioned on it.

Design Patterns

Here is a website detailing twenty two of the most common design patterns: refactoring.guru. Being able to use them to explain how you approach a code problem may also be required for a code test.

> *Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.*
>
> — *Erich Gamma*
> *Design Patterns: Elements of Reusable Object-Oriented Software*

These patterns solve extremely common problems in all software and are often extremely useful when planning out a code project in UML or trying to make your code more readable/efficient.

These patterns are the most common in my opinion

- Observer - This object is often used to watch when another object or complex system needs to change. This object may be familiar to you as the already pre implemented events system in Unreal or Unity, where the engine watches for an event to happen before running your objects code. In Web Dev an example might be a class whose responsibility is to track the position of a delivery driver to make sure they're not way off course.
- Facade *(aka: interface)* - can be *imperative* for code readability. If you have a complex collection of objects and classes that can only achieve a job together then it is a good idea to create a class to hide the complexity behind much like a library.
- Singleton - This class only ever exists once in your code. Once created any other call to create a singleton class will just return the already created instance. You can see how using this with a facade for a already running complex system like sound could reduce the amount of memory a program is using.
- Factory - Imagine you need many instances of a class to do a similar job with slightly different types. Why not make a class that manages the creation of these objects.
- Builder - So you have a highly customisable object that can be tailored to the requirements of the code. Maybe a character creator in a game, Rather than having to fill the constructor with a huge number of variables why not create a class whose sole job is to construct the object using different pieces at a time. Allowing you to break the process into steps if the default values don't need to be changed.

Over use of design patterns may just complicate your code only use them where you've noticed that it would simplify your project or will save you time and effort in the future.

UML

Is your descriptions of your solutions in your portfolio actually readable. Which type of UML should you use to describe a certain solution. Do you need to come up with a more complex visualisation to communicate what you're trying to get across. Is it readable to a stranger.

Clean Code

Readable code is imperative if someone who is trying to hire you looks at it. If they judge that your code is not worth the time to try to understand then it's unlikely that you'd be chosen over someone else whose code they can. Also if your working in a team readable code is an *extremely* important skill to stop time wasting anyway.

> *Indeed, the ratio of time spent reading versus writing is well over 10 to 1. We are constantly reading old code as part of the effort to write new code. ...*

[Therefore,] making it easy to read makes it easier to write.

— Robert C. Martin
Clean Code: A Handbook of Agile Software Craftsmanship

Key questions to ask yourself

1. Would a stranger know what this class or function does based on it's function name and/or comment description?
2. Is this code honestly best practice? (Many if statements where a switch could be used more concisely)
3. Is there too much code in one place? Can I maybe create a separate method or class to handle this strange case.
4. Is there any duplicate code? What can I do to reduce the amount of code in my program without harming the functionality.
5. Am I conforming to the correct naming schemes. Unreal uses PascalCase a lot for example where most other programs use camelCase.

Task

Analyse the code you have that'll be accessible to recruiters. Is it good enough. Can you make it good enough before they look at it?