Functional Diagram Practice & git 101

Have out Paper and Pencil *or an iPad/Tablet+Stylus* - Laptops placed away! Bags under seats!

Only use Zones ABC. No DEF!

Kris Jordan / September 6, 2024 / COMP423 / Class 07

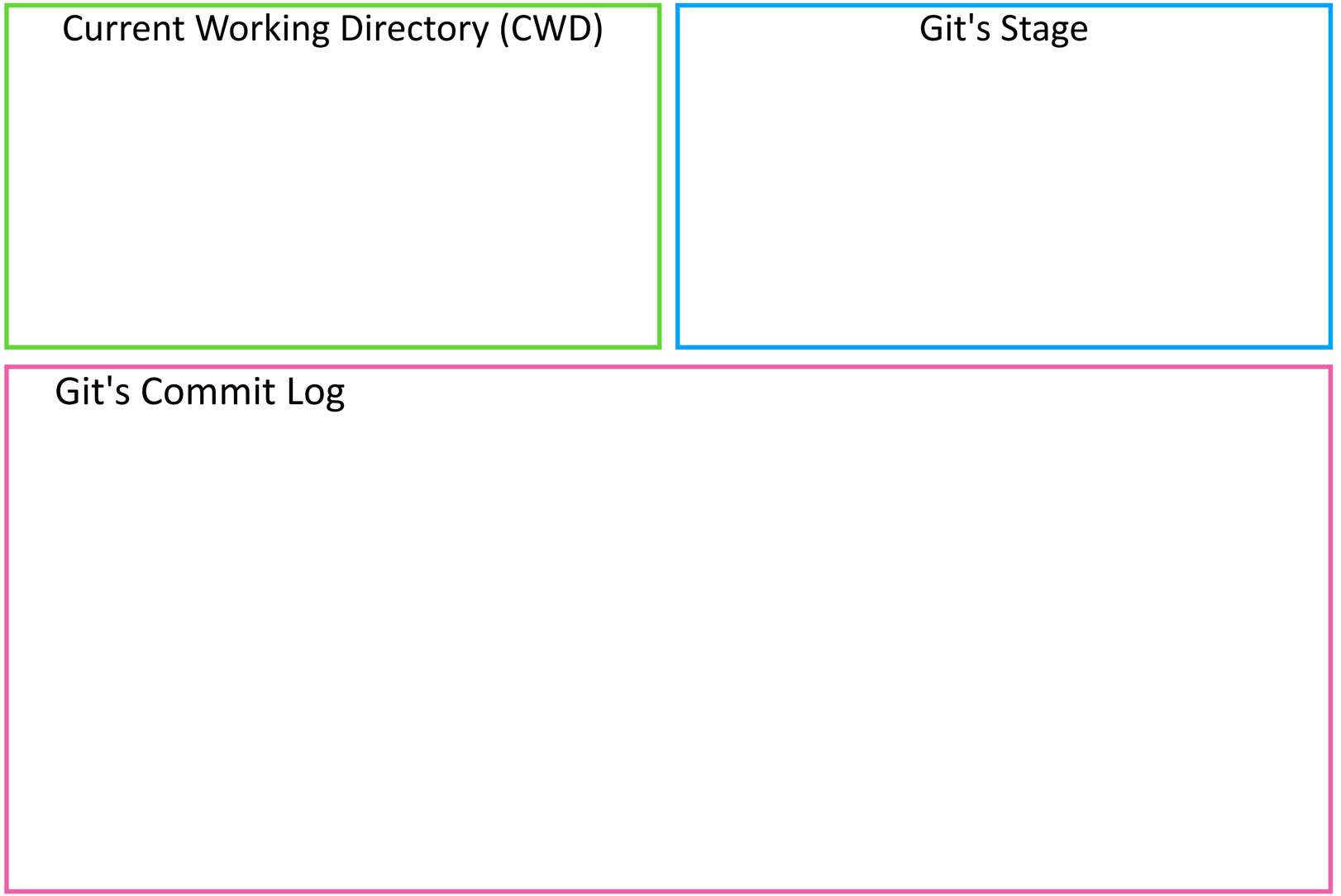
Diagram the following code listing

```
1   export const f = (x: number) => {
2     return (y: number) => {
3       return (z: number) => {
4         return x + y + z;
5       };
6     };
7   };
8
9   const a = f(1);
10  const b = a(2);
11  const c = b(3);
12  console.log(c);
```

```ts
export const f = (x: number) => {
  return (y: number) => {
    return (z: number) => {
      return x + y + z;
    };
  };
};

const a = f(1);
const b = a(2);
const c = b(3);
console.log(c);
```

```
$ git init

$ echo "hello" >demo

$ git add demo

$ git commit -m 'First commit.'
[main (root-commit) 6012aef]
 First commit.
 1 file changed, 1 insertion(+)
 create mode 100644 demo

$ echo "world" >>demo
$ git add demo

$ git commit -m 'Second commit.'
[main 773ba11] Second commit.
 1 file changed, 1 insertion(+)
```

Discuss with a partner what you believe is going on as each of the commands to the left is executed.



Quiz Topics for Wednesday

- Memory Diagramming w/ Higher-order Functions and Closures

- Learn a CLI (Shell 101 / Paths)
- Tools & DevContainer 101 (High-level concepts, not details!)
- Type Inference (*you* infer the missing type, like TypeScript would)
- Asynchronous Functions and Promises
- Unit Testing 101 (high-level concepts, e.g. what is mocking vs. instrumenting/"spying")
- Dynamically vs. Statically Typed Languages