Learning Aims

- Determine which parts of a program can be tackled at the same time
- Determine the benefits and trade-offs of concurrent processing

Thinking Concurrently

- ‘Thinking concurrently’ is the need to work out which parts of a program can be developed to be processed **at the same time**, and which parts are dependent on other parts

Thinking Concurrently

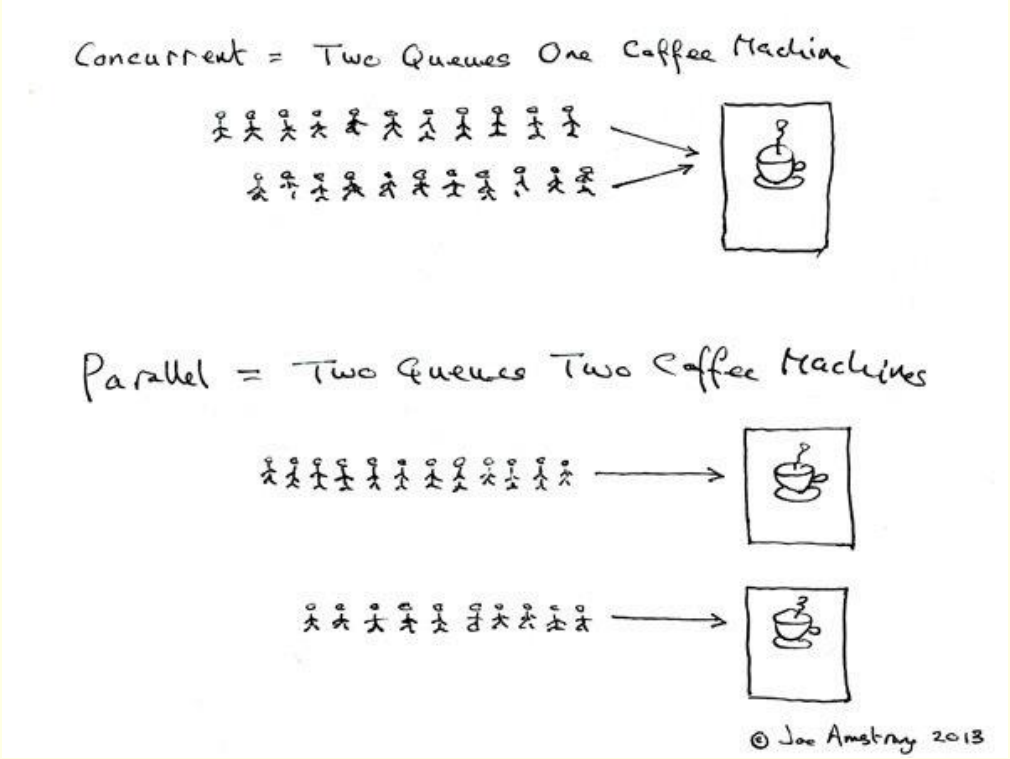
- Any situation in the design or programming of a system when you would want more than one thing happening at the same time.
- Means programs need to be specifically designed to take advantage of this.
- Modules processed at the same time should be independent.
- Well-designed programs can save a lot of processing time.
- Programs have to be written specifically to take advantage of parallel processing which can make them longer and more complex.
- However, there is the issue of sequential tasks, **not all tasks can run concurrently**, which means sometimes there is no time saving as all the tasks still need to be run in order.

Concurrent computing vs Parallel computing

- Generally, concurrent computing is defined as being related to but distinct from parallel computing.

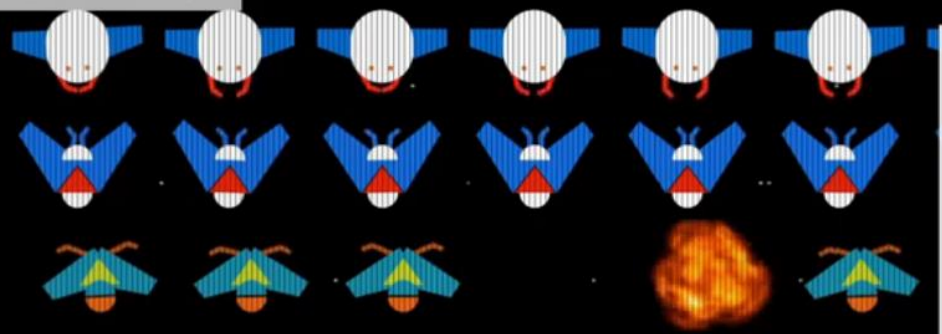
- **Parallel computing** requires **multiple processors** each executing different instructions **simultaneously**, with the goal of speeding up computations.
- It is impossible on a single processor.

- **Concurrent processing**, on the other hand, takes place when several processes are running, with each in turn being given a **slice of processor time**. This gives the appearance that several tasks are being performed simultaneously, even though only **one processor** is being used. (Processor scheduling algorithms)



Being able to think concurrently

- Moving the player ship.
- Firing bullets.
- Bullet movement.
- Enemy ship movement.
- Updating players score.
- Collision detection.

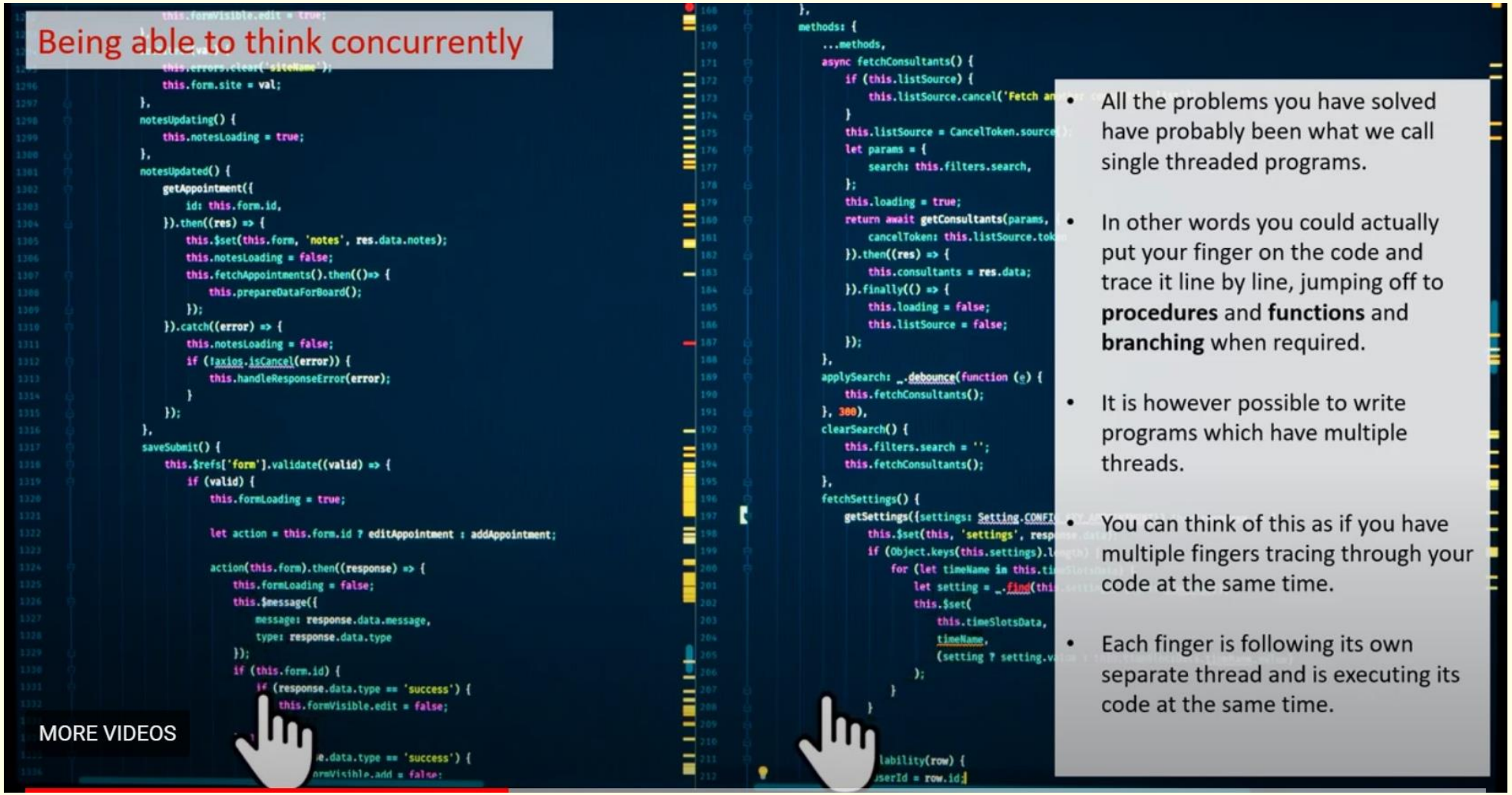


- Consider this simple space shooter game.
- **Thinking concurrently**, what aspects of this problem could be developed to take place (be processed) at the same time?

Concurrent computing and multithreading

- Parallel - Processes are happening at the same time only if multicore
- Concurrent - On a single core only 1 process can actually happen at a time, concurrent tries to **simulate multiple processes**. Processes can be in progress at the same time.
- Individual processes are **threads,** each thread has a life line.

Multi-threading



- All the problems you have solved have probably been what we call single threaded programs.
- In other words you could actually put your finger on the code and trace it line by line, jumping off to **procedures** and **functions** and **branching** when required.
- It is however possible to write programs which have multiple threads.
- You can think of this as if you have multiple fingers tracing through your code at the same time.
- Each finger is following its own separate thread and is executing its code at the same time.

Benefits and trade-offs of concurrent processing

- Concurrent processing has benefits in many situations.
- Increased program **throughput** – the number of **tasks** completed **in a given time** is increased
- Time that would be wasted by the processor **waiting** for the user to input data or look at output is used on another task
- The drawback is that If a large number of users are all trying to run programs, and some of these involve a lot of computation, these programs will take longer to complete

Benefits and trade-offs of parallel processing

- Parallel processors enable several tasks to be performed simultaneously by different processors.
- It can speed up processing enormously when repetitive calculations need to be performed on large amounts of data
- Graphics processors can quickly render a 3-D object by working simultaneously on individual components of the graphic
- A browser can display several web pages in separate windows and one processor may be carrying out a lengthy search or query while processing continues in other windows

Parallel processing has limitations:

- There is an overhead in coordinating the processors and some tasks may run faster with a single processor than with multiple processors.