Open Source Software Project Development

Dr. T.Y. Wong

Week 14

20 Minutes of Advanced Aspects in JS

- something nice in JS…

Requirement

- Problem statement:
  - I want to use XHR calls to **start downloading several files at the same time**!
  - After all downloads have been finished, invoke the callback function that I provided.

Example Usage
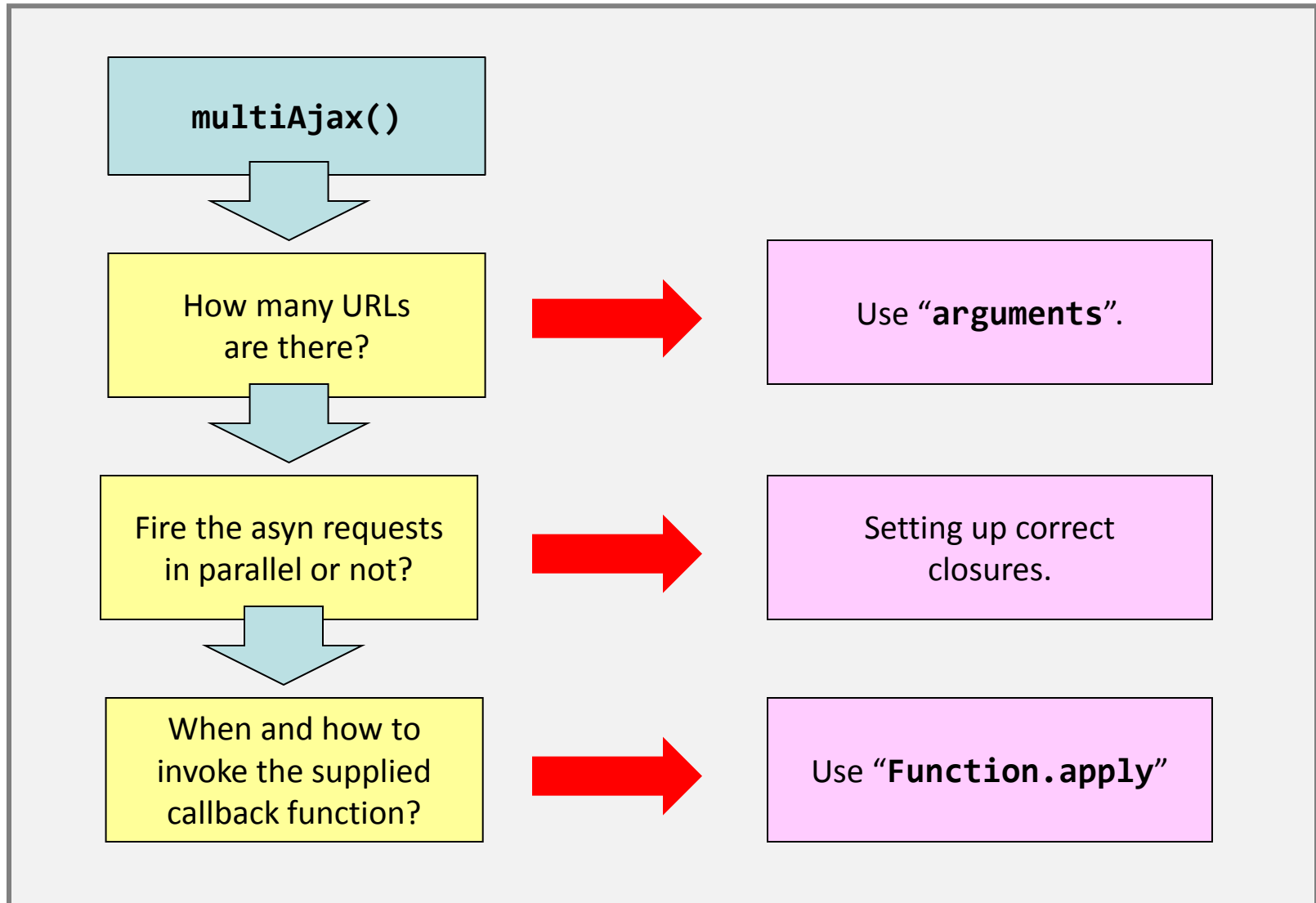
```
function callbackFunc(result_0, result_1, result_2) {
    ......
}


multiAjax(inputURL_0, inputURL_1, inputURL_2, callbackFunc);
```

Design Flow

```
multiAjax()
```

How many URLs are there? → Use "**arguments**".

Fire the asyn requests in parallel or not? → Setting up correct closures.

When and how to invoke the supplied callback function? → Use "**Function.apply**"

Challenge #1

- Argument count? Argument vector?

```
multiAjax(inputURL_0, inputURL_1, inputURL_2, callbackFunc);

multiAjax(inputURL_0, inputURL_1, callbackFunc);
```

Can I access the argument list as an array in the function?

- Here comes the "**arguments**" variable!
  - This is a default array in any functions!
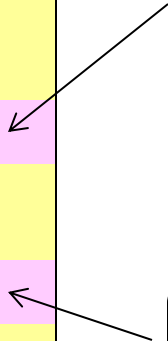 - Just like "**argv[ ]**" in C!

Challenge #1 – solved!

```
function multiAjax() {

    console.log(arguments.length);

    for(var i = 0; i < arguments.length; i++) {
        console.log(arguments[i]);
    }

}
```
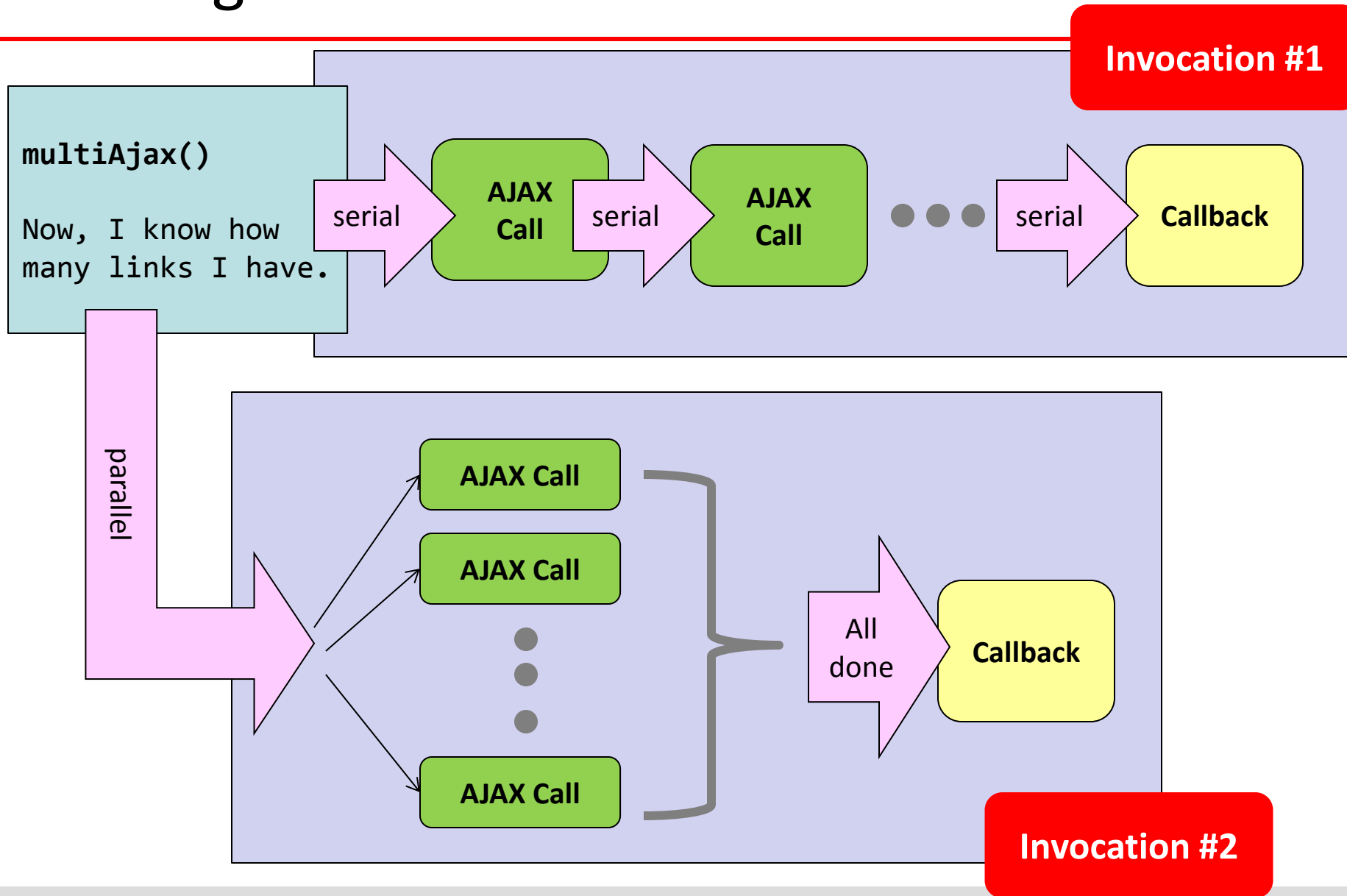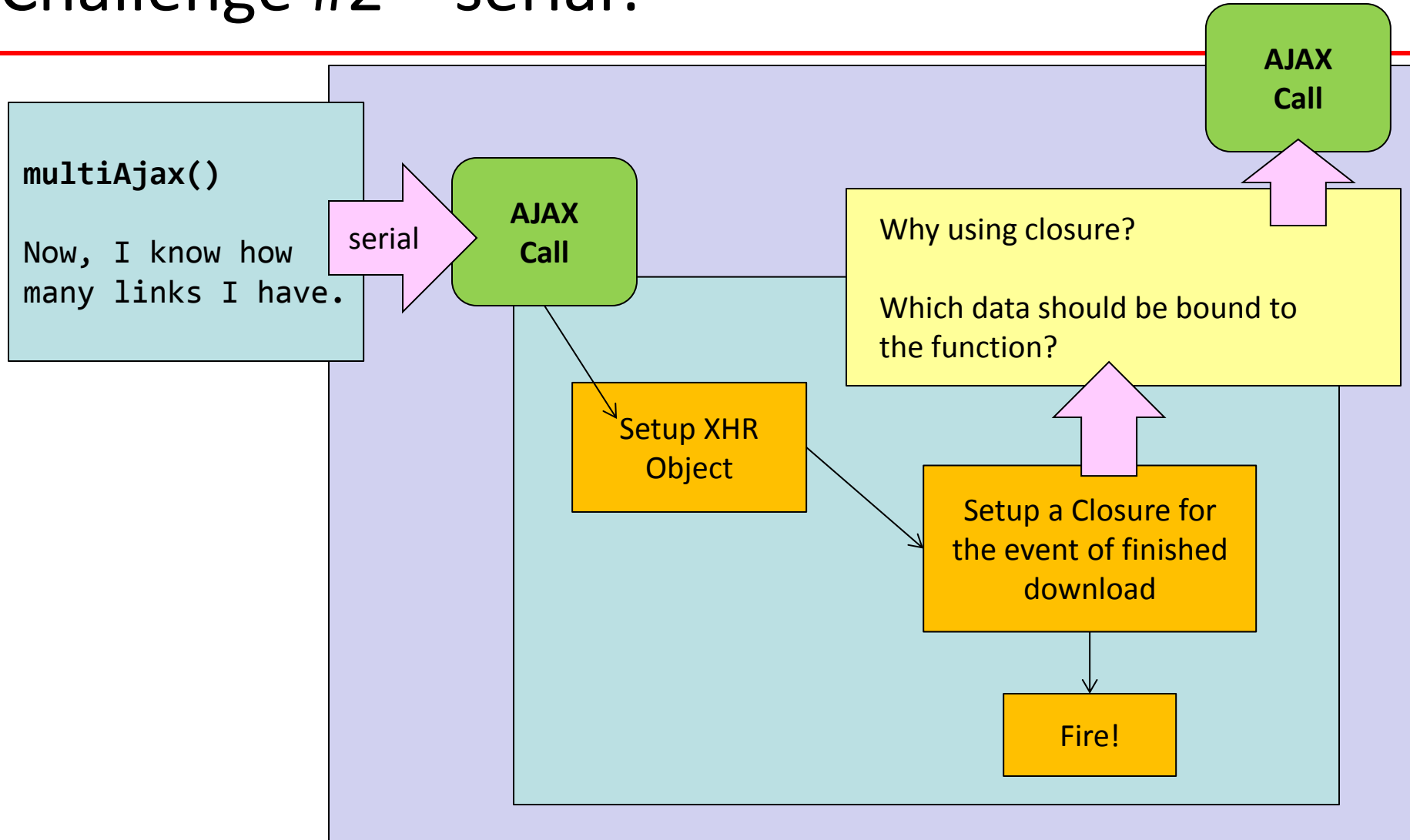
Argument Count

Argument List

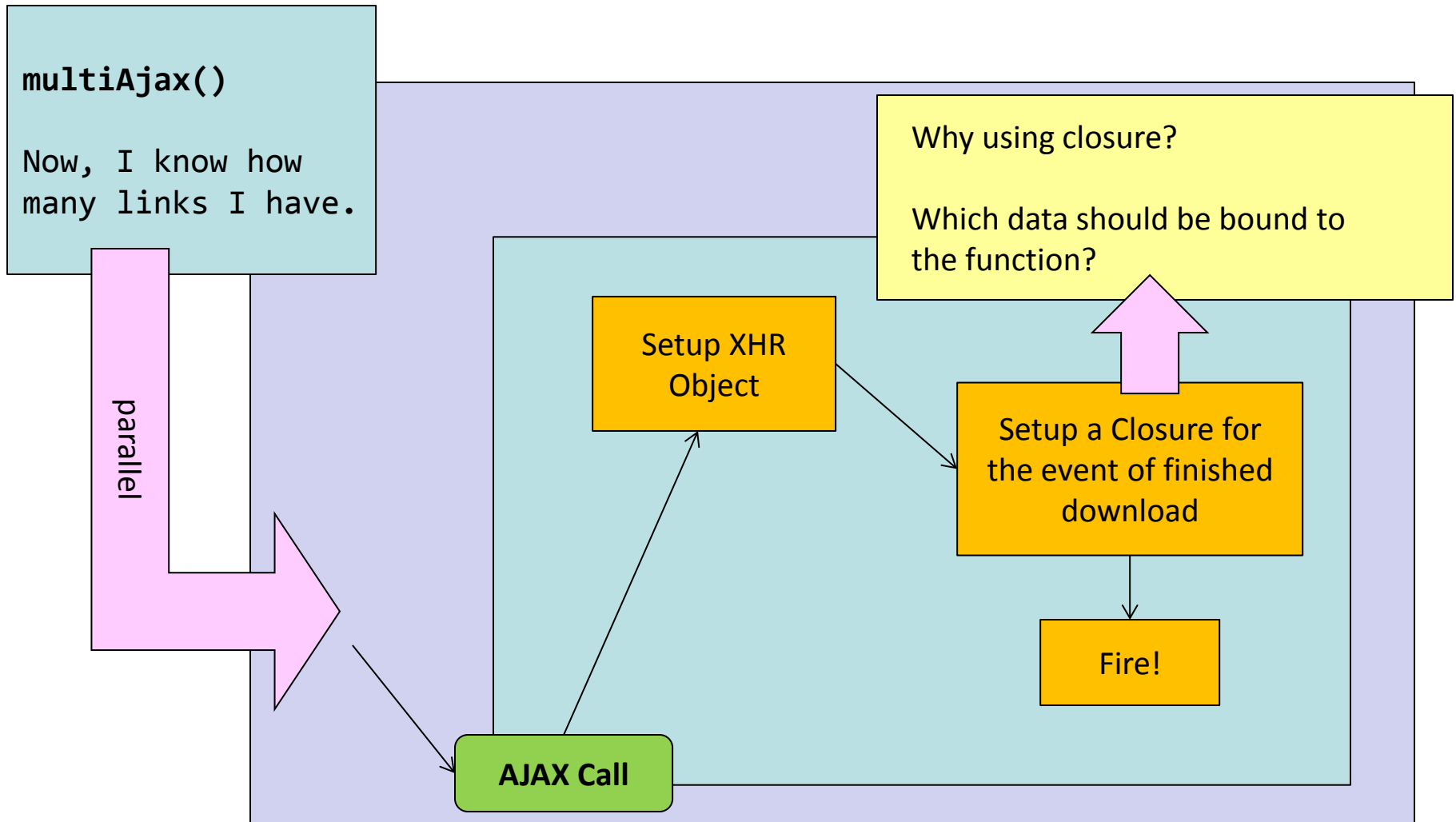**[Example]** "arguments.html", "sum.html"

Challenge #2

Challenge #2 – serial?

multiAjax()

Now, I know how many links I have.

serial →

AJAX Call

AJAX Call

Setup XHR Object

Why using closure?

Which data should be bound to the function?

Setup a Closure for the event of finished download

Fire!

[Example] "serial_multiAjax.js", "serial_download.html"

Challenge #2 – parallel?

multiAjax()

Now, I know how
many links I have.

parallel

AJAX Call

Setup XHR
Object

Setup a Closure for
the event of finished
download

Fire!

Why using closure?

Which data should be bound to
the function?

[Example] "parallel_multiAjax.js", "parallel_download.html"

Challenge #3 – final callback?

- Remember, the final callback should be called with a list of arguments. But…
  - How can we pre-defined the number of arguments?
  - How can we call without pre-defining the number of arguments?

```
function callbackFunc(result_0, result_1, result_2) {
    ......
}


multiAjax(inputURL_0, inputURL_1, inputURL_2, callbackFunc);
```

Challenge #3 – The "**apply**" pattern!

- It allows you to apply run the functions:
  - with any number of arguments, filled as an array, and
  - allowing you changing the "this" object!

- The apply pattern is everywhere in jQuery!

END