**Day 2 02 October 2023**

**Same Origin**

Same origin policy states that if we a reference to another window e.g a popup created by window.open or a window inside <iframe>, and that window comes from the same origin then that simply means we have a fully access to that window. Then if it comes from another origin, then we do not have an access to the content of that window.

**In Action: IFrame**

An <iframe> tag hosts a separate embedded window, with it own separate document and window objects.

iframe.onload vs ifrmae.contentWindow.onload

They basically the same. They triggers when the embedded window fullu loads with all resources.

But we cannot have an access iframe.contentWindow.onload for an iframe from another origin, so using iframe.onload.

**Windows on Subdomains: Document.domain**

We simply know that 2 urls that have different domains leads to different origins.

If windows share the same second level domain for example, uviwe.site.com and site.com (so their common second level domain is site.com), then we can make the browser ignore the difference and they can be treated as coming to the same origin for the purpose of cross-window communication.

**IFrame: Wrong document Pitfall**

When an iframe comes from the same origin, we have access to its document, there’s a pitfall.

How to detect the moment when the document is there?

When the iframe.onload triggers already the right document is there. But it only triggers when the whole iframe with resources is loaded.

**Collection: Window.iframes**

Another way of getting a window object for <iframes> is to get from the named collectionwindow.frames

* By number: window.frames[0] = the window object for the first frame in the document.
* By name: window.frames.iframeName = the window object for the frame withname= “iframeName”.

Navigation links are:

* window.frames – the collection of “children” windows (for nested frames).
* window.parent – the reference to the “parent” (outer) window.
* window.top – the reference to the topmost parent window.

**The *Sandbox* iframe attribute**

Sandbox allows for exclusion of certain actions inside <iframe> in order to prevent it from executing untrusted code. It *sandboxes* the iframe by treating it as coming from another different origin and applying other limitations.

The purpose of the sandbox attribute is only to add more restrictions. It cannot remove them.

**Cross window messaging**

postMessage interface is the one that allows windows to talk to each other no matter which origin the comes from.

The interface contains 2 parts:

postMassage:

A window that wants to send a message calls postMessage of receiving window. This simply means if the we want to send the message to win. We should call win.postMessage(data, targetOrigin).

Arguments:

Arguments has kind of parts which are.

Data:

The data to send. Can be any object.

targetOrigin:

Specifies the origin for the target window, so that only a window from the origin will get the message.

Specifying the targetOrigin ensures that the window only receives the data if its still at the right site. Import when the data is sensitive.

onmessage.

To receive a message, the target window should have a handler on the message event. It triggers when postMassage is called (and targetOrigin check is successful).