Cross window messages

The `postMessage` interface allows two windows with any origins to talk:

1. The sender calls `targetWin.postMessage(data, targetOrigin)`.
2. If `targetOrigin` is not `'*'`, then the browser checks if window `targetWin` has the origin `targetOrigin`.
3. If it is so, then `targetWin` triggers the `message` event with special properties:

- `origin` – the origin of the sender window (like `http://my.site.com`)
- `source` – the reference to the sender window.
- `data` – the data, any object in everywhere except IE that supports only strings.

The clickjacking attack

The "clickjacking" attack allows an evil page to click on a "victim site" *on behalf of the visitor*.

Clickjacking is a way to "trick" users into clicking on a victim site without even knowing what's happening. That's dangerous if there are important click-activated actions.

A hacker can post a link to their evil page in a message, or lure visitors to their page by some other means. There are many variations.

From one perspective – the attack is "not deep": all a hacker is doing is intercepting a single click. But from another perspective, if the hacker knows that after the click another control will appear, then they may use cunning messages to coerce the user into clicking on them as well.

The attack is quite dangerous, because when we engineer the UI we usually don't anticipate that a hacker may click on behalf of the visitor. So vulnerabilities can be found in totally unexpected places.

- It is recommended to use `X-Frame-Options: SAMEORIGIN` on pages (or whole websites) which are not intended to be viewed inside frames.
- Use a covering `<div>` if we want to allow our pages to be shown in iframes, but still stay safe.

X-FRAME OPTIONS

The server-side header `X-Frame-Options` can permit or forbid displaying the page inside a frame.

It must be sent exactly as HTTP-header: the browser will ignore it if found in HTML `<meta>` tag. So, `<meta http-equiv="X-Frame-Options"...>` won't do anything.

The header may have 3 values:

`DENY`

Never ever show the page inside a frame.
`SAMEORIGIN`

Allow inside a frame if the parent document comes from the same origin.
`ALLOW-FROM domain`

Allow inside a frame if the parent document is from the given domain.
For instance, Twitter uses `X-Frame-Options: SAMEORIGIN`.

ARRAYBUFFER

`ArrayBuffer` is the core object, a reference to the fixed-length contiguous memory area.

To do almost any operation on `ArrayBuffer`, we need a view.

- It can be a `TypedArray`:
- `Uint8Array`, `Uint16Array`, `Uint32Array` – for unsigned integers of 8, 16, and 32 bits.
- `Uint8ClampedArray` – for 8-bit integers, "clamps" them on assignment.
- `Int8Array`, `Int16Array`, `Int32Array` – for signed integer numbers (can be negative).
- `Float32Array`, `Float64Array` – for signed floating-point numbers of 32 and 64 bits.
- Or a `DataView` – the view that uses methods to specify a format, e.g. `getUint8(offset)`.

In most cases we create and operate directly on typed arrays, leaving `ArrayBuffer` under cover, as a "common denominator". We can access it as `.buffer` and make another view if needed.

There are also two additional terms, that are used in descriptions of methods that operate on binary data:

- `ArrayBufferView` is an umbrella term for all these kinds of views.
- `BufferSource` is an umbrella term for `ArrayBuffer` or `ArrayBufferView`.

We’ll see these terms in the next chapters. `BufferSource` is one of the most common terms, as it means “any kind of binary data” – an `ArrayBuffer` or a view over it.