

Inspire.js

Lean, hackable, extensible
slide deck framework

By Lea Verou and contributors

Introduction

Main idea

Main idea

- An HTML file contains the whole presentation

Main idea

- An HTML file contains the whole presentation
- Themes as CSS files

Main idea

- An HTML file contains the whole presentation
- Themes as CSS files
- Plugins for extra functionality, autoloaded when used

Main idea

- An HTML file contains the whole presentation
- Themes as CSS files
- Plugins for extra functionality, autoloaded when used
- Nice JavaScript API for really custom things















History

- Originally created in 2010 for my first talk, at Front Trends 2010
- By popular demand, it was released shortly after, named CSSS
- Huge refactor and renaming to Inspire.js in Sep 2018

Syntax

- Slides are defined by adding the **slide** class to any elements.
- `<header class="slide">` is a header slide (relevant for the Overview)

Keyboard Navigation

-  or  to advance to the next slide or incrementally displayed item
-  or  to go to the previous slide or incrementally displayed item
-   or   to jump to the next slide
-   or   to jump to the previous slide
- Home to go to the first slide, End to go to the last
-   to jump to an arbitrary slide (by slide number or identifier)

Features

Repeated slides

- Use attribute **data-insert="#id"** on an empty slide to re-insert it later in the slideshow
- Notice how this slide will also be re-inserted after the next one!

IDs and titles

- IDs are dynamically assigned by JavaScript, but you can also use your own
- Slide titles (used in the browser tab when the slide is active, and in the Overview) via the **title** or **data-title** attributes or headings in the slide.
- If only an id or only a title is provided, the other is extracted from it.

Repeated slides

- Use attribute **data-insert="#id"** on an empty slide to re-insert it later in the slideshow
- Notice how this slide will also be re-inserted after the next one!

Incremental display

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items
- Use selector `.delayed.displayed` to change the style of displayed items

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items
- Use selector `.delayed.displayed` to change the style of displayed items
- Use selector `.delayed.current` to change the style of current items

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items
- Use selector `.delayed.displayed` to change the style of displayed items
- Use selector `.delayed.current` to change the style of current items
- Use `class="delayed-children"` to apply the **delayed** class to all of an element's children

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items
- Use selector `.delayed.displayed` to change the style of displayed items
- Use selector `.delayed.current` to change the style of current items
- Use `class="delayed-children"` to apply the `delayed` class to all of an element's children
- Nested delayed items !

Incremental display

- Incremental display of slide contents (just add `class="delayed"`)
- Use selector `.delayed:not(.current):not(.displayed)` to change the style of undisplayed items
- Use selector `.delayed.displayed` to change the style of displayed items
- Use selector `.delayed.current` to change the style of current items
- Use `class="delayed-children"` to apply the `delayed` class to all of an element's children
- Nested delayed items **are possible!**

Plugins

Overview

- Loaded automatically, use `<code class="no-overview">` to prevent this.
- Press `Ctrl` + `H` (or `Shift` + `H`) to trigger, `Esc` to close.

Overview

- Loaded automatically, use `<code class="no-overview">` to prevent this.
- Press `Ctrl` + `H` (or `Shift` + `H`) to trigger, `Esc` to close.
- Cool, huh? You can press `Ctrl` `Shift` `H` to see all slides

- Autoloads core & languages used
- Load plugins by listing their ids in a **data-prism-plugins** attribute, on any element (first one will be used).
- Automatically figures out dependencies & aliases
- CSS is up to you!

```
start = new Date().getTime();
setInterval(function() {
  react.render(
    <ExampleApplication elapsed={new Date().getTime() - start} />
    document.getElementById('container')
  );
};
```