Home | Download / Install | Tutorials | Live examples | Documentation | Forum | Source

Knockout.

Simplify dynamic JavaScript UIs with the Model-View-View Model (MVVM) pattern

Download
v3.4.2 - 25kb min+gz

release notes

Key concepts

Declarative Bindings
Easily associate DOM elements with model data using a concise, readable syntax

Automatic UI Refresh
When your data model's state changes, your UI updates automatically

Dependency Tracking
Implicitly set up chains of relationships between model data, to transform and combine it

Templating
Quickly generate sophisticated, nested UIs as a function of your model data

More features

- ✓ Free, open source (MIT license)
- ✓ Pure JavaScript — works with any web framework
- ✓ Small & lightweight — 59kb minified
  ... reduces to 25kb when using HTTP compression
- ✓ No dependencies
- ✓ Supports all mainstream browsers, even ancient ones
  IE 6+, Firefox 3.5+, Chrome, Opera, Safari (desktop/mobile)
- ✓ Fully documented
  API docs, live examples, and interactive tutorials included

Get started

Interactive tutorials
Learn the easy way with an in-browser code editor

20-minute demo video

Grab a coffee and watch this fast-paced introductory session, filmed at the 2011 MIX conference.

Introduction to Knockout
Tutorial, benefits, comparisons with other frameworks

PluralSight Knockout.js training course
Nearly 5 hours of online video content by John Papa (more info)

Introduction to the Model-View-View Model pattern
How KO enables it with observables and computed properties

More live examples

External links and blog posts

Source code on Github

New: Interactive tutorials

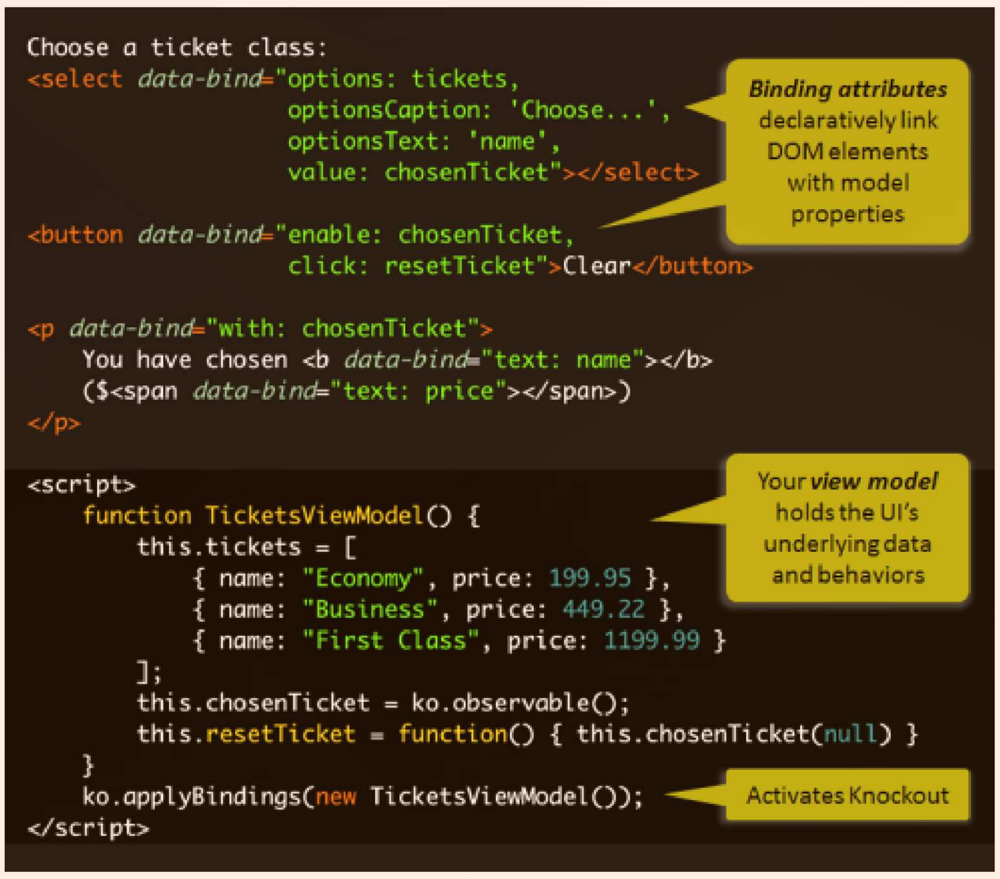
Get started with knockout.js quickly, learning to build *single-page applications*, *custom bindings* and more with these interactive tutorials.

Live example

Run it:

Choose a ticket class: Choose... ▾ Clear

Source code:



```
Choose a ticket class:
<select data-bind="options: tickets,
                   optionsCaption: 'Choose...',
                   optionsText: 'name',
                   value: chosenTicket"></select>

<button data-bind="enable: chosenTicket,
                   click: resetTicket">Clear</button>

<p data-bind="with: chosenTicket">
    You have chosen <b data-bind="text: name"></b>
    ($<span data-bind="text: price"></span>)
</p>

<script>
    function TicketsViewModel() {
        this.tickets = [
            { name: "Economy", price: 199.95 },
            { name: "Business", price: 449.22 },
            { name: "First Class", price: 1199.99 }
        ];
        this.chosenTicket = ko.observable();
        this.resetTicket = function() { this.chosenTicket(null) }
    }
    ko.applyBindings(new TicketsViewModel());
</script>
```

Binding attributes declaratively link DOM elements with model properties
Your view model holds the UI's underlying data and behaviors
Activates Knockout

By encapsulating data and behavior into a view model, you get a clean, extensible foundation on which to build sophisticated UIs without getting lost in a tangle of event handlers and manual DOM updates.