Knockout.

IN THE ENGINE ROOM

Mike Geyser
@mikegeyser

JSINSA
June 2013

What is KnockoutJS?

- **Model-View-View Model (MVVM)**
  - **Model:** Data, domain. and state
    (Web server. Database.)
  - **View:** Display.
    (*Html. CSS.)*
  - **View Model:** Data, operations, behaviour.
    (Javascript + Data bindings.)

What is KnockoutJS?

- **Headline features:**
  - **Elegant dependency tracking:**
    - Observable pattern, subscriptions.
  - **Declarative bindings:**
    - Data-bind attribute.
  - **Trivially extensible:**
    - Just plain old javascript.

Wait, what is this talk actually about?

- Model-View-View Model (MVVM)
  - *Model*
  - *View*
  - ***View Model***
- Headline features:
  1. **Elegant dependency tracking**
  2. Declarative bindings
  3. Trivially extensible

Wait, what is this talk actually about?

Part 1: Subscriptions basics

- Explicit manual subscriptions
- Bindings
- Computed observables
- Dependency chain

DEMO

Part 1: Subscriptions basics

- **Every observable access is a subscription**:
  - Manual, data-bind, computed.

- **Lots of little pieces**
  - Good way to build complex SPAs
  - Can quickly escalate – leading to performance problems.

"The secret to building large apps is never build large apps. Break your applications into small pieces. Then, assemble those testable, bite-sized pieces into your big application"

- Justin Meyer, author JavaScriptMVC

Part 2: Array handling

- Foreach template binding
- Array manipulation
- Controlling subscription notifications

DEMO

Part 2: Array handling

- **Naïve usage:**
  - Potential for lots of callbacks.
  - Certain performance problems.
- **Manual notification:**
  - Explicitly control notification.
  - Ability to have granular control.
  - Embrace the reference semantics.
- **Contrived example:**
  - AJAX data loading. Multiple insert/delete.

Part 3: Custom binding handler

- Foreach and non-trivial template markup
- Computed result sets
- Custom binding handler

DEMO

Part 3: Custom binding handler

- **Templating:**
  - Can be expensive, even if model manipulation if efficient.
  - UI **will** become complex.

- **Custom binding handlers:**
  - Not just for reuse.
  - Separation of concerns.
  - Keeping markup out of the model.

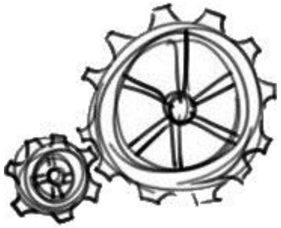
Well? Is that it?

- **Knockout can be performant:**
  - Not trivial.
  - Not hacky.
  - Solid approach and foundation.
- **More:**
  - extends({}), fn, utils, custom observables.
  - Plain old javascript.
  - Still have a server.

Thanks!

https://twitter.com/mikegeyser

For comments, criticisms, rude gestures and cat pictures.



https://jabbr.net/#/rooms/knockoutjs

For great help, heated argument, and :trollface:

https://github.com/mikegeyser/knockout-in-the-engine-room

For full code, notes, instructions and references.