SPA - View

View

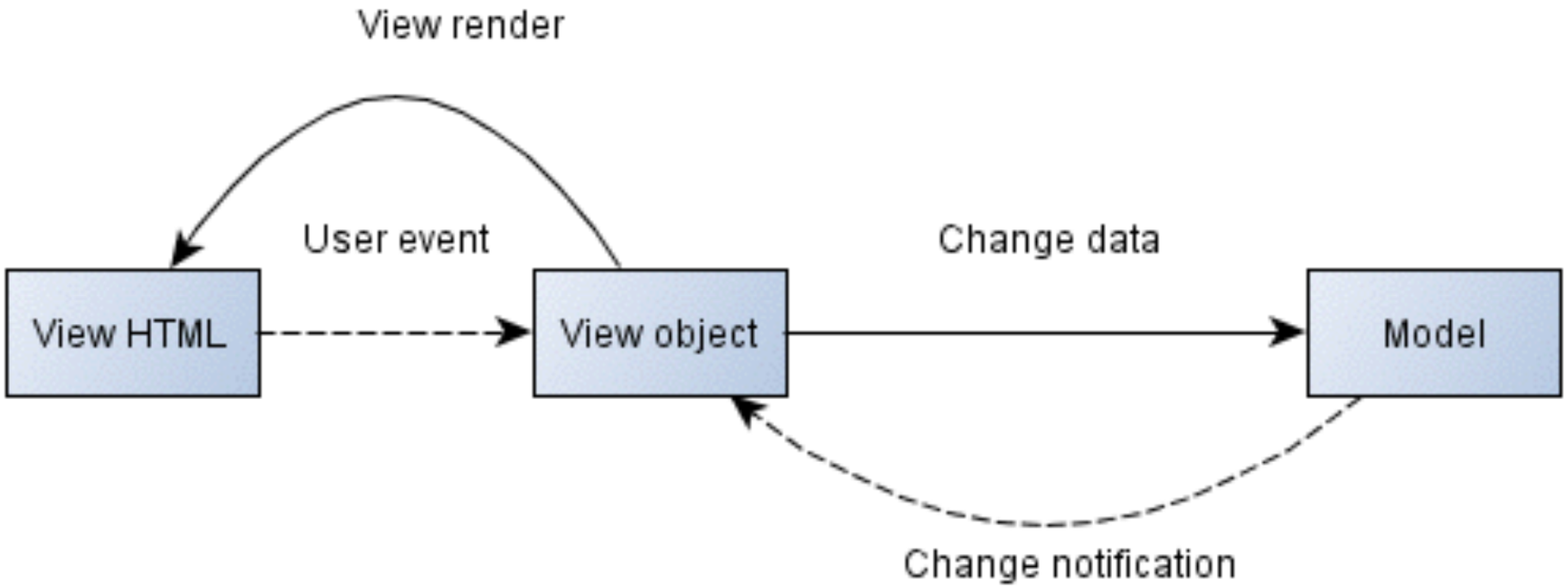
Views have several tasks to care of:

- Rendering a template.

- Updating views in response to change events

- Binding behaviour to HTML via event handlers.

View


View render
User event
Change data
View HTML
View object
Model
Change notification

View

There are two questions:

How should event handlers be bound to/unbound from HTML?

At what granularity should data updates be performed?

View Layer dimensions/contrasts

1. Low end interactivity vs. high end interactivity
2. Close to server vs. close to client
3. Markup-driven views vs. Model-backed view

Low-end interactivity vs High-end interactivity

Low-end interactivity

- Example: Github
- Pages are mostly static.
- You take a document that represents a mostly static piece of information already processed, and add a bit of interactivity via Javascript.
- Changing data usually causes a full page refresh.

High-end interactivity

- Example: Gmail
- Pages are mostly dynamic.
- You have a set of data which you want the user to interact with in various ways; changes to the data should be reflected on the page immediately.
- Changing data should update the views, but not cause a page refresh - because views have many small intermediate states which are not stored in the database.

- State and data can be stored in HTML, because if data is altered, the page is refreshed.
- Because a majority of the state is in the HTML, parts of the UI do not generally interact with each other.
- If complex interactions are needed, they are resolved on the server.

- Storing state and data in HTML is a bad idea, because it makes it hard to keep multiple views that represent the same data in sync.
- Complex interactions are more feasible; data is separate from presentation.
- Interactions don't need to map to backend actions, e.g. you can paginate and select items in a view without writing a new server-side endpoint.

Close to server vs. Close to client

Server generates HTML

HTML manipulation

PJAX

Server as JSON API

Markup-driven views

Model-backed views

Widgets

Markup-driven views vs Model-backed views

[Data in JS models] [Data in JS models]

[Model-backed views] [Markup accesses models]

Model-backed views

```
var model = new Todo({ title: 'foo', done: false }),
view = new TodoView(model);
```

Markup-driven views

```
{{view TodoView}}

  {{=window.model.title}}

{{/view}}
```