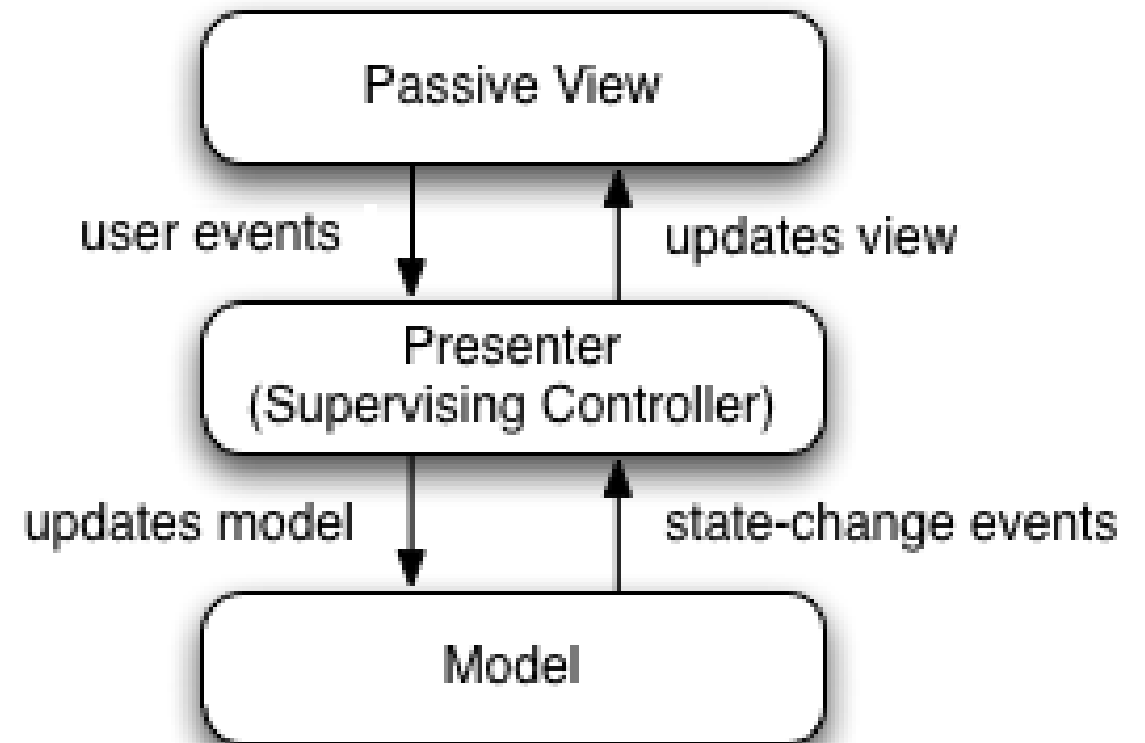# Building UIs with Spec

- Model View Presenter
- Object Oriented

The MVP pattern

- Model
- View
- Presenter

Defining a presenter

- Subclass `SpPresenter`

- Often 1 instance variable per internal wideget

- Overrides methods:
  - `#defaultLayout`
  - `#initializePresenters`
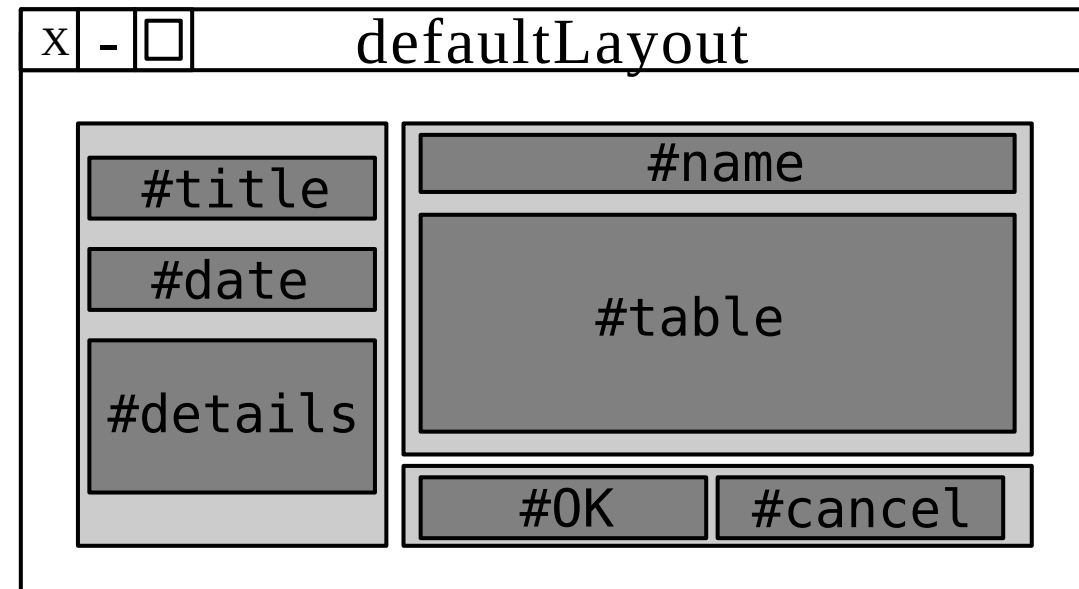 - `#connectPresenter`

#initializePresenters

```
initializePresenters
  title := self newDropList .
  name := self newDropList .
  date := self newTextInput .
  details := self newText.
  table := self newTable.
  ok := self newButton
    label:#ok;
      yourself.
  cancel := self newButton
    label:#cancel;
      yourself.
```

#defaultLayout

```
defaultLayout
  ^ SpBoxLayout newLeftToRight
      add: (SpBoxLayout newTopToBottom
        add: #title
          height: self toolbarHeight;
        add: #date
          height: self toolbarHeight;
        add: #details);
      add: (SpBoxLayout newTopToBottom
        add: #name
          height: self toolbarHeight;
        add: #table;
        add: (SpBoxLayout newLeftToRight
          add: #ok;
          add: #cancel));
      yourself
```

#connectPresenters

```
connectPresenters
  ok action: [
    self inform:'Ok'
  ].
  cancel action: [
    self inform:'Cancel'
  ].
```

connectPresenters

| | v | | v |

01/01/1970

| | | | |

ok   cancel

Binding the model

```
model: aModel
    model := aModel.
    self modelChanged.
```
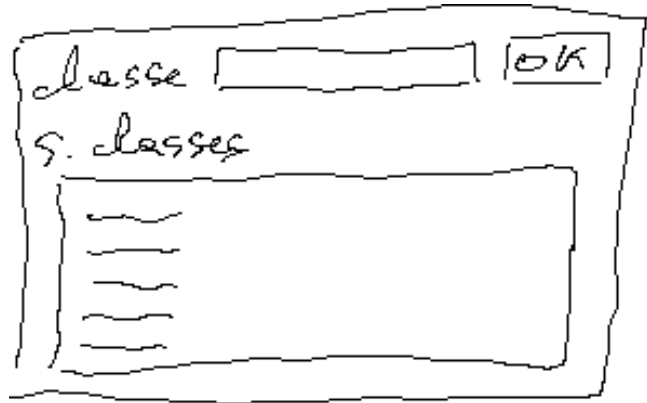
```
modelChanged
    self fillWidgetWithModel
```

```
fillWidgetWithModel
    title selectedItem: model title.
    name selectedItem: model name.
    date text: model birthDate.
    details text: model details.
```

Exercice

- Créer l'interface suivante

- Entrer un nom de classe dans le champ texte et cliquer sur "OK" pour afficher la liste de ses sous-classes

- version avancée: les sous-classes sont dans une liste hiérarchique montrant l'héritage entre elles