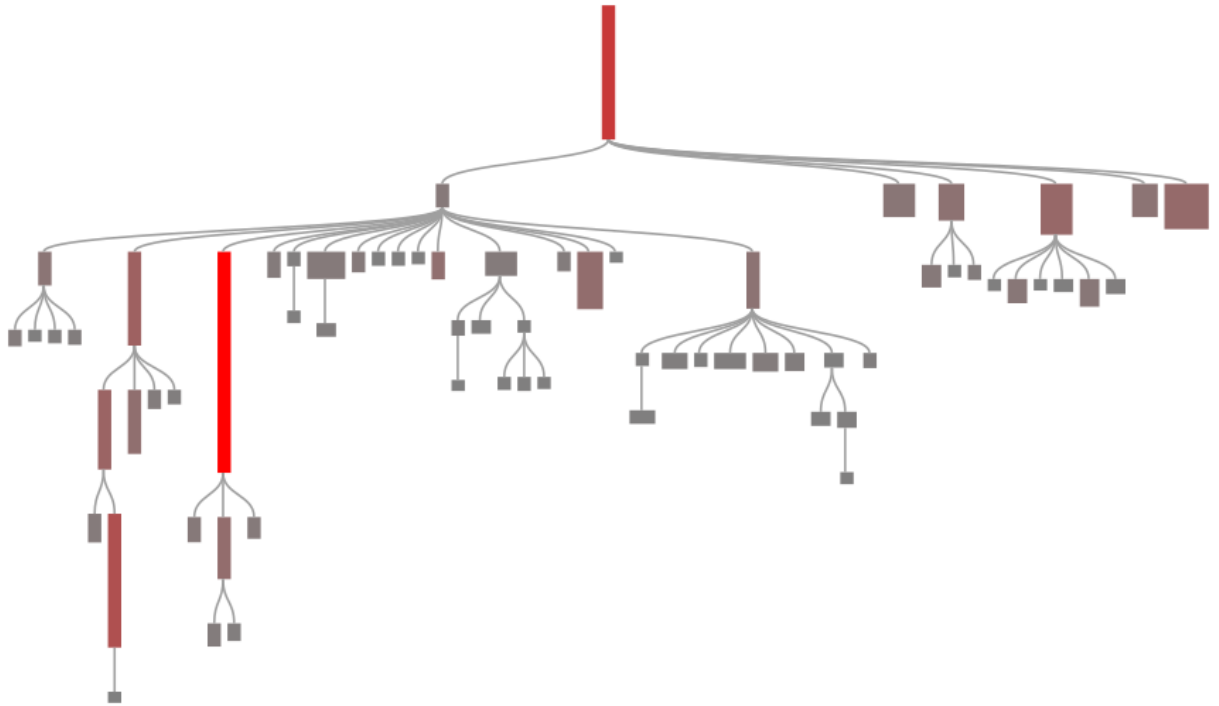
Roassal



Roassal

- Visualisation de données
- Un ensemble d'outils
- Dessiner des formes
- Disposer des formes
- Interactions avec les formes

Exemples

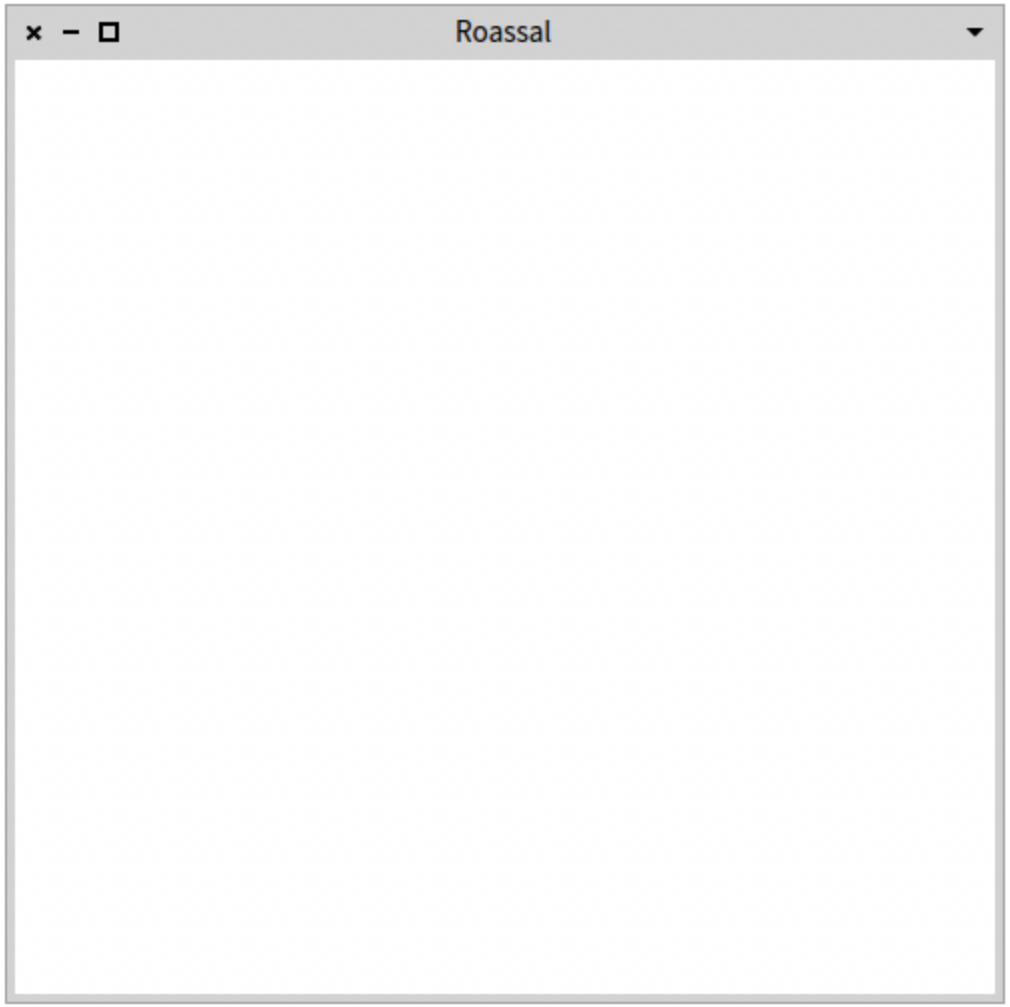
Roassal - Compsants principaux

- Canvas
- Shapes
- Layouts
- Events
- Interactions

Le canvas

- Contient et affiche les formes

```
canvas := RSCanvas new.
canvas open
```



Les formes

- Sous classes de `RSShape`
  - Rectangle : `RSBox`
  - Cercle : `RSCircle`
  - Ligne : `RSLine`
  - Texte : `RSLabel`
  - etc.

Les formes

- Rectangle

```
rect := RSBox new.
```

- Cercle

```
circle := RSCircle new.
```

Moduler les formes

Propriétés : `#height:` , `#width:` , couleur, bordure, couleur

```
rect height: 100;
     width: 50;
     color: Color red.
```

Les formes

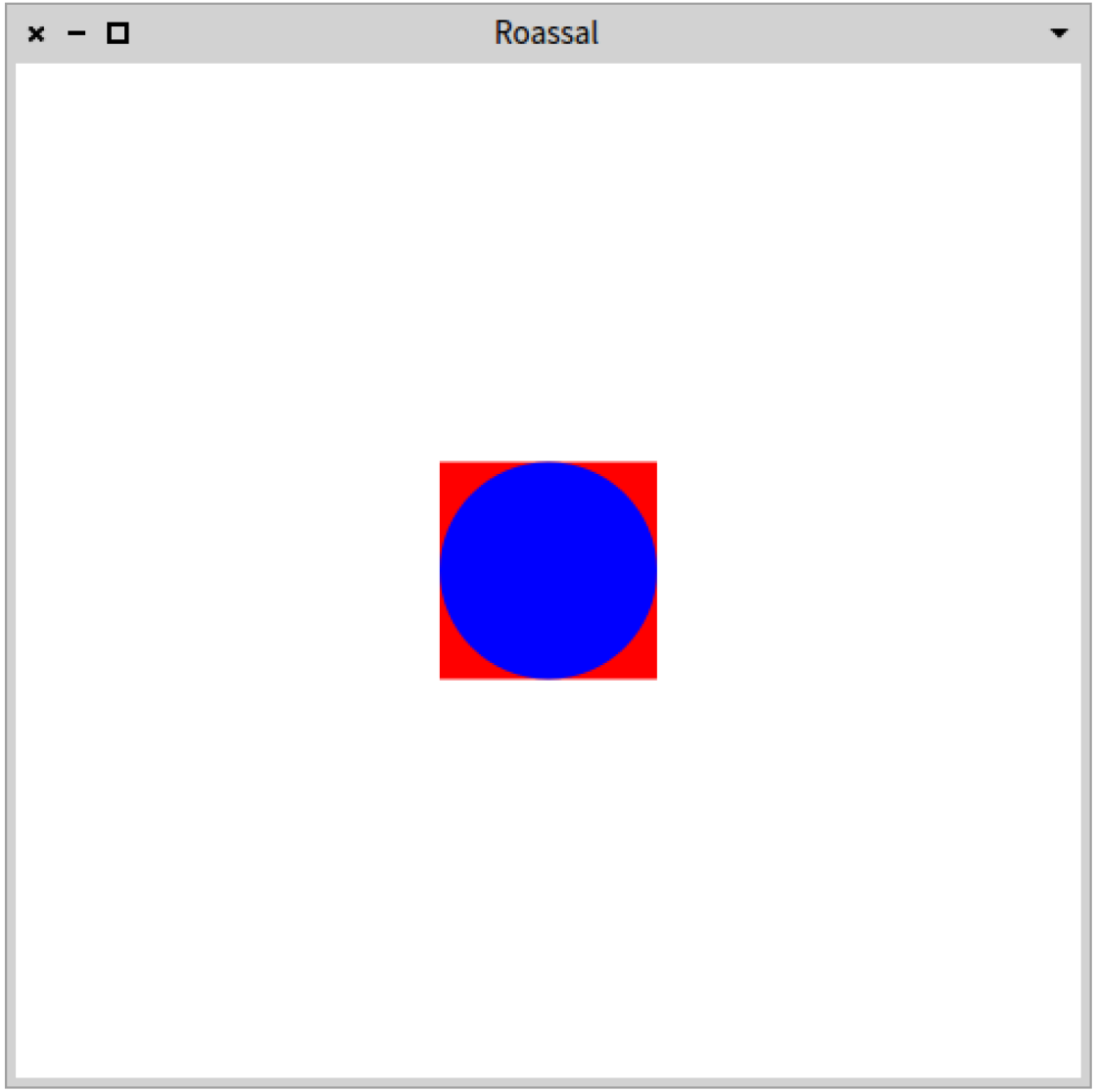
- Associer une donnée utilisateur (modèle) aux formes
  - Une forme peut représenter un object Pharo
  - Actions sur la forme en fonction de l'objet représenté

```
shape model: 1
```

Les formes dans le canvas

- Rectangle

```
canvas add: rect.
canvas add: circle.
canvas open
```



Application

Pour chaque classe de la hiérarchie de la classe XXX, **collecter** un rectangle qui la décrit .

Le résultat doit être un ensemble de formes.

Ajouter ces formes dans un canvas et ouvrir le canvas.

Les layouts

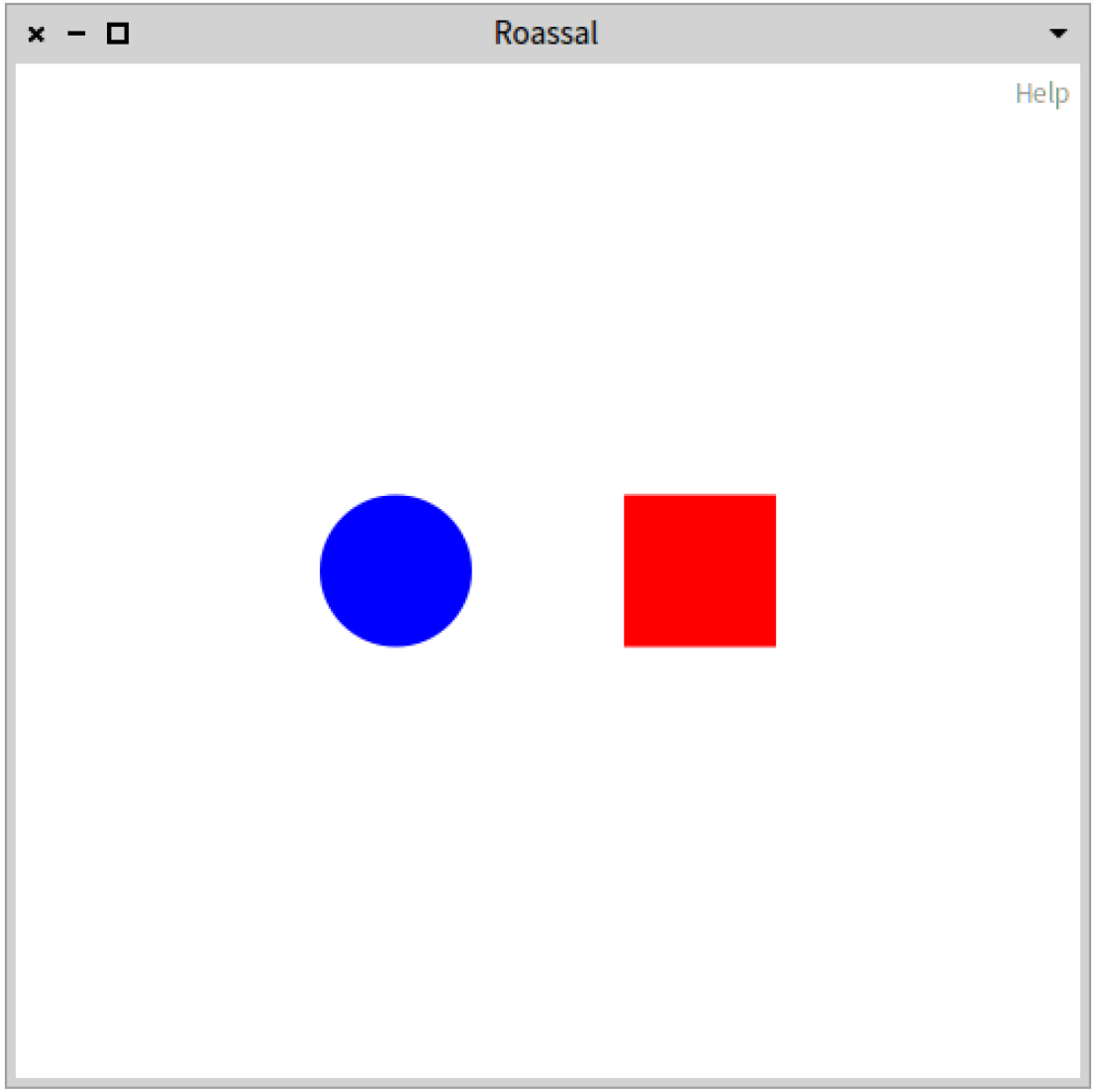
- Permettent de gérer la disposition des objets sur le canvas
  - Disposition horizontale `RSHorizontalLineLayout`
  - Disposition verticale `RSVerticalLineLayout`
  - Disposition arborescente `RSTreeLayout`
  - etc.

Les layouts

- Disposition horizontale

  `RSHorizontalLineLayout`

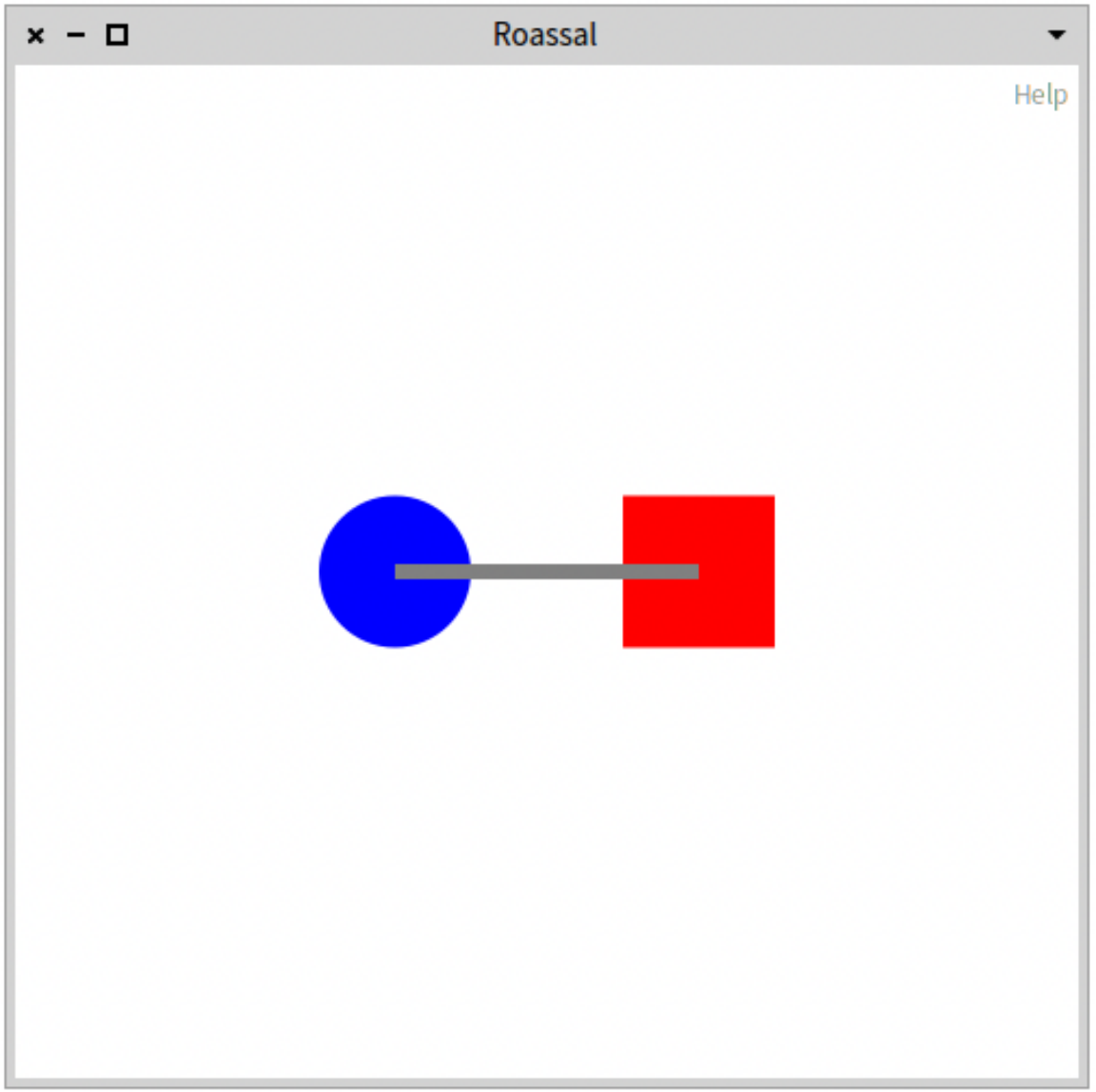
```
RSHorizontalLineLayout on: {circle, rect}.
canvas add: circle;
       add: rect.
canvas open
```



Les liens

- Lier des formes

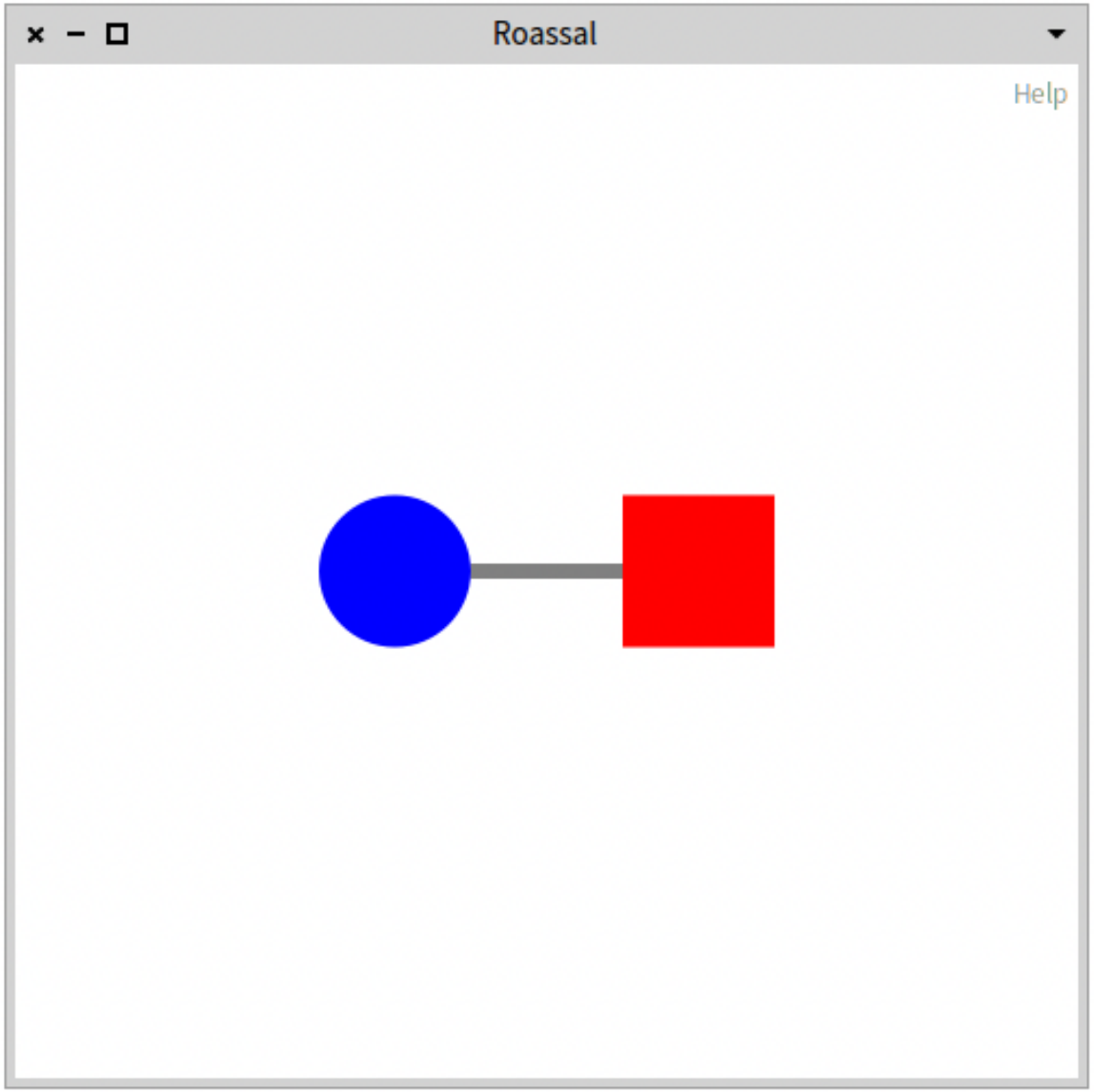
```
line := RSLine new.
line from: rect;
     to: circle.
canvas add: line.
```



Les liens

- Avec un point d'attache différent

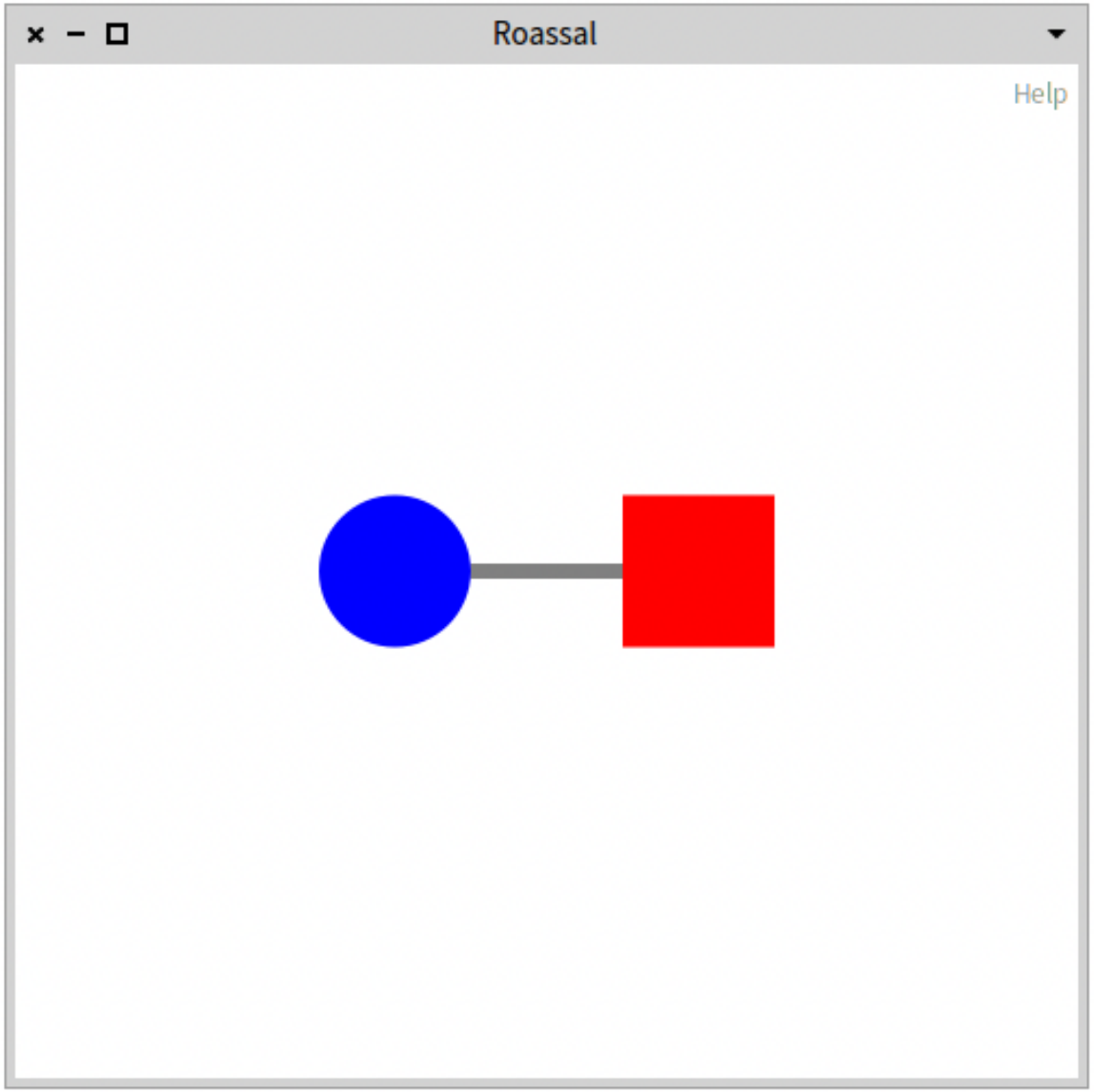
```
line := RSLine new.
line withBorderAttachPoint;
     from: rect;
     to: circle.
canvas add: line.
```



Les liens

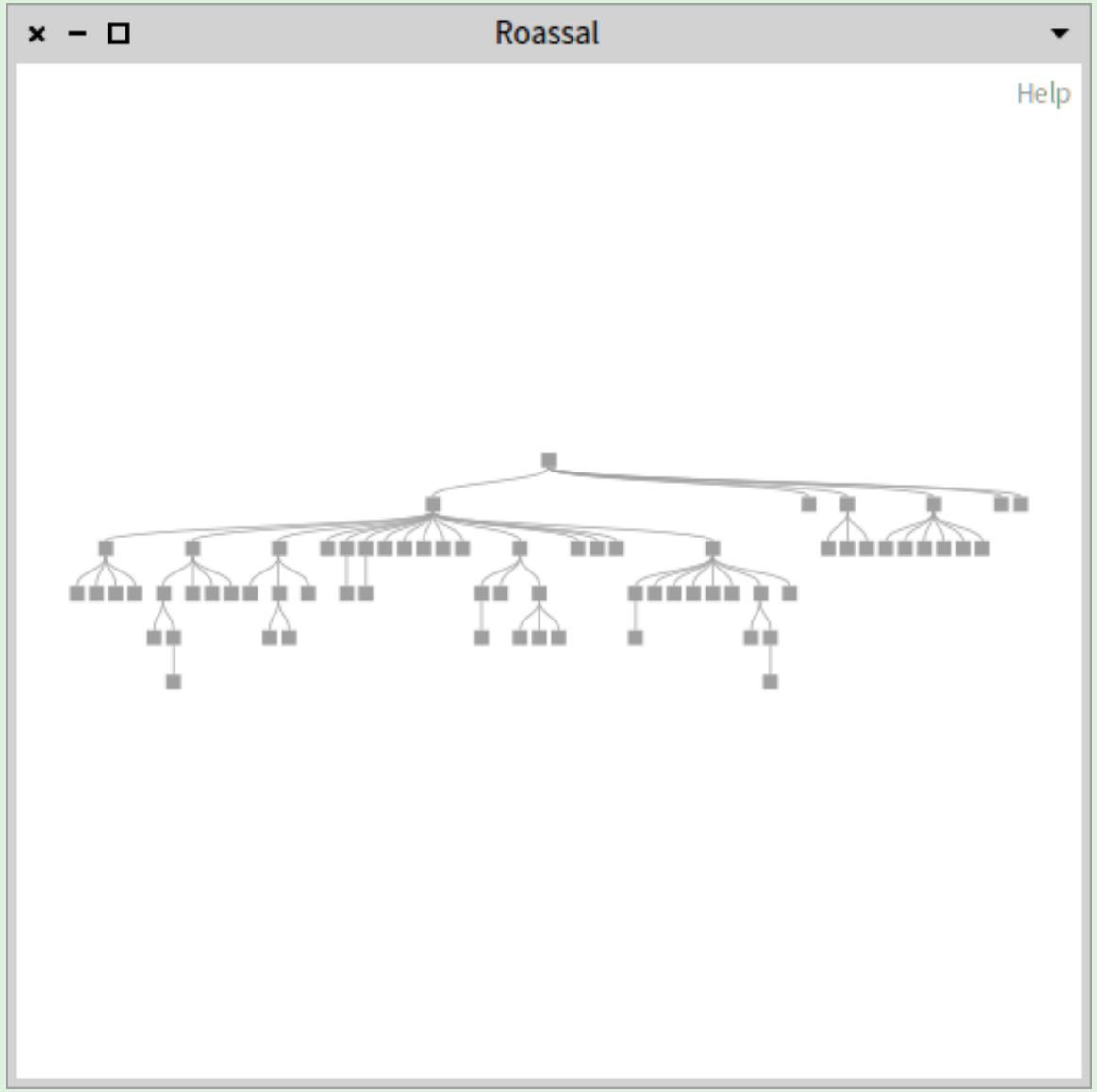
- Avec un builder

```
RSLineBuilder line
        canvas: c;
        connectFrom: #aSelectorOnTheModel.
```



Application

Créer des liens entre les classes et leur super classe.
Disposer les classe de façon à obtenir une arborescence.



Les évènements

- Sous classes de RSEvent.
  - `RSMouseClick`, `RSMouseEnter`, `RSKeyDown`, etc.

```
shape on: RSMouseClick do: [ :evt | "Action à réaliser" ]
```

Les interactions

- Draggable `RSDraggable`
- Popup `RSPopup`
- Highlight `RSHighlightable`
- Menu `RSMenuActivable`

```
shape @ RSPopup "Affiche le nom du modèle au passage de la souris"
```

Application

Ajouter une interaction de votre choix sur les classes.
Ajouter un évènement sur chaque forme, permettant d'inspecter son modèle avec un clic droit.

Quelques outils Roassal

- Normalizer

```
RSNormalizer height
  shapes: c shapes;
  normalize: #numberOfMethods.
```

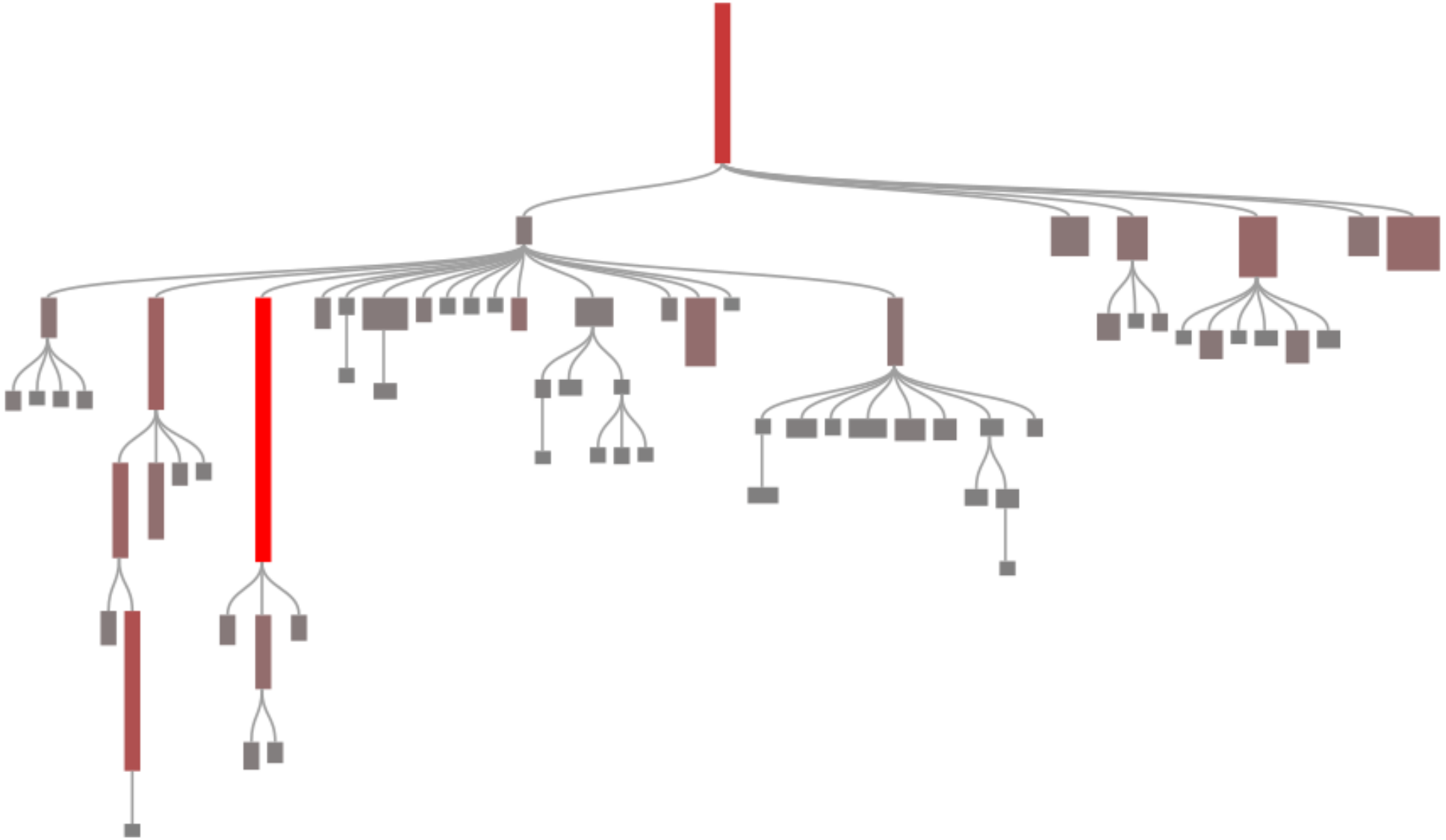
- Exporters (pdf, svg, png, ...)

```
RSPNGExporter new
                  canvas: self;
    filname: 'myCanvas';
    export
```

Application

Modifier votre visualisation afin d'adapter la taille des classes en fonction de leurs propriétés :

- La hauteur : nombre de méthodes de la classe
- La largeur : nombre d'attributs de la classe
- La couleur : nombre de ligne de code



Ressources

- Github (MIT)
  - https://github.com/ObjectProfile/Roassal3
- Documentation
  - https://github.com/ObjectProfile/Roassal3Documentation:
- Agile Visualization
  - http://agilevisualization.com/