

# Xcode Interface Builder

Ceci est un fichier XML au format XIB (Xcode Interface Builder), utilisé pour définir l'interface utilisateur d'une application macOS écrite en Cocoa. Plus précisément, ce fichier XIB décrit une fenêtre de paramètres pour une application appelée "Reveal-In-GitHub." Laissez-moi vous le décomposer étape par étape :

## Vue d'ensemble

Ce fichier définit une fenêtre intitulée "Reveal-In-GitHub Settings" avec divers éléments d'interface utilisateur comme des boutons, des champs de texte et une vue personnalisée. Il est conçu à l'aide de l'Interface Builder (partie de Xcode) et utilise Auto Layout pour positionner les éléments de manière dynamique. La fenêtre est contrôlée par une classe personnalisée appelée `RIGSettingWindowController`.

## Sections clés

### 1. Informations d'en-tête

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="com.apple.InterfaceBuilder3.Cocoa.XIB" version="3.0" toolsVersion="9059" systemVersion="10.14.6" targetRuntime="MacOSX" colorMatched="true" lastOpenFile=""></document>
```

- Cela spécifie qu'il s'agit d'un fichier XIB pour une application Cocoa macOS, utilisant la version 3.0 du format XIB.
- `toolsVersion="9059"` et `systemVersion="14F27"` indiquent les versions de Xcode et de macOS utilisées lors de sa création.
- `useAutolayout="YES"` signifie que la mise en page s'ajuste dynamiquement en fonction de la taille de la fenêtre ou de la résolution de l'écran.

### 2. Dépendances

```
<dependencies>
  <plugIn identifier="com.apple.InterfaceBuilder.CocoaPlugin" version="9059"/>
</dependencies>
```

- Cela fait référence au plugin Cocoa pour Interface Builder, qui fournit le cadre pour les éléments d'interface utilisateur macOS.

### 3. Objets

Cette section définit tous les objets de l'interface, y compris la fenêtre, les vues, les boutons et leurs connexions.

- **Propriétaire du fichier (`RIGSettingWindowController`)**

```
<customObject id="-2" userLabel="File's Owner" customClass="RIGSettingWindowController">
  <connections>
    <outlet property="configsView" destination="IKd-Ev-B9V" id="z2z-MF-G88"/>
  </connections>
</customObject>
```

```
<outlet property="mainView" destination="se5-gp-Tj0" id="pGn-EC-mzi"/>
<outlet property="window" destination="F0z-JX-Cv5" id="gIp-Ho-8D9"/>
</connections>
</customObject>
```

- Le Propriétaire du fichier est la classe de contrôleur (`RIGSettingWindowController`) qui gère cette fenêtre.
  - Il possède des sorties (connexions) vers la vue principale, une vue de configurations et la fenêtre elle-même, reliant les éléments de l'interface utilisateur au code.

- Fenêtre

```
<window title="Reveal-In-GitHub Settings" ... id="F0z-JX-Cv5" userLabel="Settings">
    <>windowStyleMask key="styleMask" titled="YES" closable="YES" miniaturizable="YES" resizable="YES"
    <rect key="contentRect" x="527" y="176" width="651" height="497"/>
    <rect key="screenRect" x="0.0" y="0.0" width="1440" height="877"/>
```

- Définit une fenêtre avec un titre, des boutons de fermeture/minimisation/redimensionnement, et une taille de 651x497 pixels.
  - Le `contentRect` spécifie sa position et sa taille sur l'écran, tandis que `screenRect` décrit la résolution complète de l'écran (1440x877).

- **Vue de contenu**

```
<view key="contentView" id="se5-gp-Tj0">
    <rect key="frame" x="0.0" y="0.0" width="651" height="497"/>
    <subviews>
```

- La vue de contenu principale remplit la fenêtre et contient tous les éléments de l'interface utilisateur (sous-vues).

4. **Éléments de l'interface utilisateur (sous-vues)** Voici quelques éléments clés à l'intérieur de la vue de contenu :

- **Champs de texte**

```
<textField ... id="G1C-Td-n9Y">  
    <rect key="frame" x="18" y="425" width="187" height="17"/>  
    <textFieldCell ... title="Custom Menu Items" ...>
```

- Une étiquette indiquant “Custom Menu Items” à la position (18, 425).
  - D’autres champs de texte incluent “Menu Title,” “^ ↑ ☰ +” (raccourci Control+Shift+Command), et “URL Pattern.”

- **Boutons**

```
<button ... id="EuN-9g-Vcg">  
  <rect key="frame" x="14" y="13" width="137" height="32"/>  
  <buttonCell ... title="Save" ...>
```

```

<connections>
    <action selector="saveButtonClicked:" target="-2" id="efd-QE-4aQ"/>
</connections>

```

- Un bouton “Save” à (14, 13) qui déclenche la méthode saveButtonClicked: dans le contrôleur.
- D’autres boutons incluent “Clear Default Repos” et “Reset Default Menus,” chacun lié à des actions spécifiques.

- **Vue personnalisée**

```

<customView ... id="IKd-Ev-B9V" userLabel="Config View">
    <rect key="frame" x="20" y="54" width="611" height="330"/>

```

- Un espace réservé pour une vue personnalisée (probablement peuplée de manière programmatique) prenant la majeure partie de la fenêtre.

## 5. Connexions

- Les boutons sont câblés à des méthodes dans le `RIGSettingWindowController` (par exemple, `saveButtonClicked:`, `clearButtonClicked:`), permettant aux interactions utilisateur de déclencher du code.

### À quoi cela sert-il ?

Ce fichier XIB crée une fenêtre de paramètres pour une application macOS appelée “Reveal-In-GitHub,” probablement un outil pour intégrer la fonctionnalité GitHub dans une autre application. La fenêtre permet aux utilisateurs de :

- Personnaliser les éléments de menu (avec des titres et des motifs d’URL).
- Enregistrer leurs paramètres.
- Effacer les dépôts par défaut ou réinitialiser les menus aux valeurs par défaut.

La mise en page est fixe (`fixedFrame="YES"`) plutôt que d’utiliser pleinement les contraintes Auto Layout, ce qui signifie que les positions sont codées en dur en pixels. Cela pourrait la rendre moins flexible sur différents tailles d’écran mais plus simple à concevoir.

### Pourquoi utiliser XIB ?

Les fichiers XIB permettent aux développeurs de concevoir visuellement des interfaces dans l’Interface Builder de Xcode, les connectant au code via des sorties et des actions. Ce fichier fait le lien entre l’interface utilisateur (boutons, texte, etc.) et la classe `RIGSettingWindowController`, qui gère la logique.