



Utilisation du Storyboard avec Xcode

1. Création et Navigation dans le Storyboard

1.1 Ouvrir le Storyboard

- Ouvre Xcode et accède au fichier `Main.storyboard` dans le navigateur de projet.
- L'interface affiche les vues disponibles dans ton application.



1.2 Ajouter un ViewController

- Depuis la bibliothèque d'objets (`Cmd + Shift + L`), recherche `ViewController`.
- Fais glisser un `ViewController` sur le Storyboard.
- Associe-le à une classe `UIViewController` dans l'Inspecteur d'identité.



2. Gestion des Contraintes Auto Layout

2.1 Ajouter des Contraintes

- Sélectionner un élément (Label, Button, etc.).
- Cliquer sur le bouton **Add Constraints** en bas à droite.
- Définir les marges (égales, supérieures, inférieures, etc.).

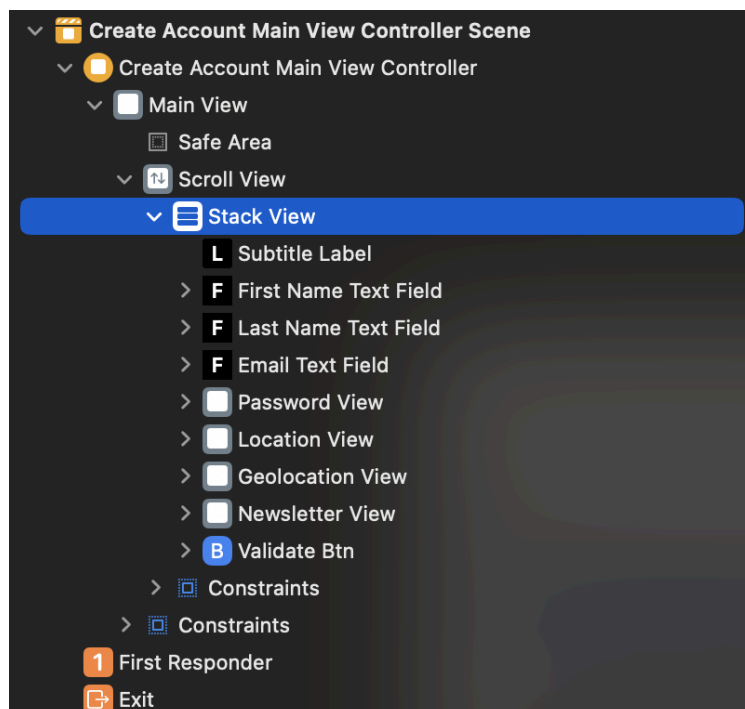
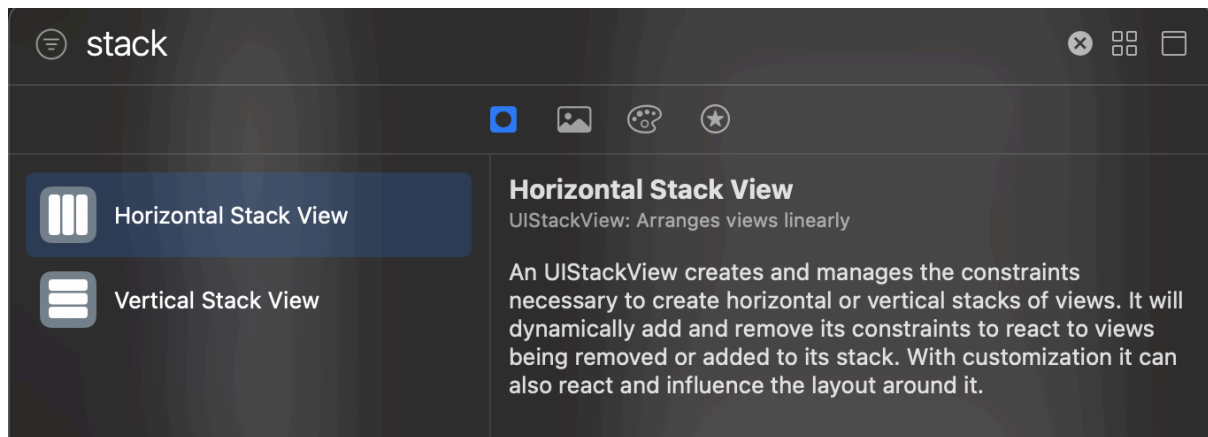
The screenshot displays the Xcode Auto Layout interface. On the left, the 'Constraints' panel shows a grid of constraints for a view. The 'Horizontal' section includes 'Trailing Space to: Superview', 'Leading Space to: Superview', and 'Equal Width to: Superview'. The 'Vertical' section includes 'Bottom Space to: Superview' and 'Top Space to: Superview'. On the right, the 'Add New Constraints' dialog is open, showing a diagram of a square with numerical values for width (343), height (570.5), and spacing to nearest neighbor (44.5). The 'Add Constraints' button is visible at the bottom of the dialog.



3. Utilisation de StackView

3.1 Ajouter une StackView

- Depuis la bibliothèque d'objets, ajouter une **StackView**.
- Configurer l'axe (**horizontal** ou **vertical**).
- Ajuster **Spacing** et **Alignment**.

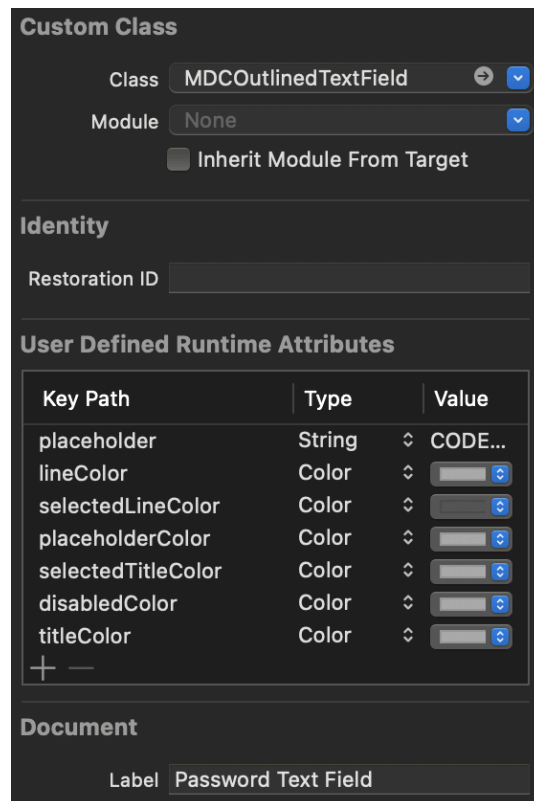




4. Gestion des Champs de Texte

4.1 Ajouter un `MDCOutlinedTextField`

- Depuis la bibliothèque d'objets, rechercher `Text Field`.
- Appliquer les contraintes pour une bonne mise en page.



4.2 Ajouter un Bouton à Droite

- Utiliser `trailingView` pour éviter le chevauchement du texte :

```
textField.trailingView = button  
textField.trailingViewMode = .always
```





5. Connexions entre Interface et Code (IBOutlets & IBActions)

5.1 Créer un IBOutlet

- Sélectionner un élément UI.
- Ctrl + Drag vers le fichier `.swift` correspondant.
- Sélectionner `Outlet` et nommer la variable.

```
20 // MARK: - Outlets
21
22 @IBOutlet weak var subtitleLabel: UILabel!
23 @IBOutlet weak var scrollView: UIScrollView!
24 @IBOutlet weak var stackView: UIStackView!
25 @IBOutlet weak var passwordView: UIView!
26 @IBOutlet weak var locationView: UIView!
27
28
29
30 @IBOutlet weak var firstNameTextField: MDCOutlinedTextField!
31 @IBOutlet weak var lastNameTextField: MDCOutlinedTextField!
32 @IBOutlet weak var emailTextField: MDCOutlinedTextField!
33 @IBOutlet weak var passwordTextField: MDCOutlinedTextField!
34 @IBOutlet weak var locationTextField: MDCOutlinedTextField!
```

5.2 Créer une IBAction (qui se déclenche au clic d'un bouton)

- Sélectionner un bouton.
- Ctrl + Drag vers le fichier `.swift`.
- Choisir `Action` et nommer la méthode.

```
@IBAction func passwordVisibilityButtonTapped(_ sender: Any) {
    passwordTextField.isSecureTextEntry = !passwordTextField.isSecureTextEntry
    let buttonTextKey = passwordTextField.isSecureTextEntry ?
        "CreateAccountPasswordViewController.showPassword" :
        "CreateAccountPasswordViewController.hidePassword"
}
```