

CREATION D'UN PORTAIL KIIWAY POUR LANCER UN JEU



CONVERSION EN APK

- Il est possible de convertir votre jeu **buildbox** en **apk** via le logiciel **Android Studio**
- Il est nécessaire dans un premier temps d'enregistrer votre jeu **buildbox** Pour **Google Play Store** en Version **Android**
- Après l'avoir enregistré il faudra l'ouvrir avec le logiciel **Android Studio** et aller sur l'onglet **Build**, **Generate signed Bundle**.

CREATION D'UN PORTAIL KIIWAY

- Il est nécessaire pour la création de ce portail d'utiliser le logiciel **Android Studio** qui nous permettra de créer mais aussi d'automatiser ce portail
- Ce portail vous permettra de prendre les informations des personnes qui veulent jouer à votre jeu **buildbox**

ASPECT DESIGN DU PORTAIL

- Après avoir créer un nouveau projet sur **Android Studio** il va falloir choisir ce que vous souhaitez mettre dans votre portail à partir du fichier **xml** et ceci grâce aux fonctions que vous retrouverez à gauche de la fenêtre de votre projet
- Pour notre programme on utilisera principalement les fonctions qu'on utilise pour un formulaire basique **Button**, **PlainText**, **Postal Address**, **E-mail**, **Phone**

PROGRAMMATION PORTAIL KIIWAY

- Pour programmer ce portail nous allons utiliser la partie java qui va nous permettre d'automatiser chaque fonction du portail mais aussi de définir des actions qui vont nous permettre de communiquer avec d'autres applications ou logiciel.
- Nous allons d'abord utiliser la fonction *EditText* qui va nous permettre de definir nos fonctions qu'on a créer dans le fichier **xml** comme étant du texte et aussi la fonction *button* qui va nous permettre de definir notre bouton sur le fichier **xml** comme étant un bouton. Après avoir défini les fonctions on va maintenant créer des variables à partir de ces fonctions qui nous permettront de les utiliser dans notre programme et ceci grâce à la fonction *FindViewById*

PROGRAMME DEFINIR FONCTION POUR LES OBJETS DANS LE XML

```
private EditText UserName, UserLastName, UserPadresse, UserEmail, Userphone;  
private Button StrButton;  
  
private void setupUIViews() {  
  
    UserName = (EditText) findViewById(R.id.editText15);  
    UserLastName = (EditText) findViewById(R.id.editText14);  
    UserPadresse = (EditText) findViewById(R.id.editText11);  
    UserEmail = (EditText) findViewById(R.id.editText12);  
    Userphone = (EditText) findViewById(R.id.editText6);  
    StrButton = (Button) findViewById(R.id.button);  
}
```

- Après avoir créer les variables, on va définir des conditions qui vont nous permettre d'empêcher l'utilisateur du portail de ne pas accéder au jeu que s'il rempli parfaitement le formulaire et que le mail soit envoyé.
- Pour arriver à cet objectif nous allons utiliser la fonction **boolean** et **string**, mais aussi **Toast.MakeText** qui nous permettra de définir un message d'erreur qui s'affichera quand l'utilisateur ne rempli pas toutes les parties du formulaire.

PROGRAMME CONDITION D'ENVOI D'E-MAIL

```
private Boolean validate()
{
    Boolean result = false;
    String name = UserName.getText().toString();
    String Lastname = UserLastName.getText().toString();
    String Padresse = UserPadresse.getText().toString();
    String Email = UserEmail.getText().toString();
    String Phone = Userphone.getText().toString();

    if (name.isEmpty() || Lastname.isEmpty() || Padresse.isEmpty() || Email.isEmpty() ||
    Phone.isEmpty()) {
        Toast.makeText(this, "Veuillez entrer toutes les informations",
        Toast.LENGTH_SHORT).show();
    }
    else
    {
        result = true;
    }

    return result;
}
```

- Après avoir programmer la première condition, nous allons programmer le bouton pour lui définir une ou plusieurs actions quand on clique dessus
- Pour arriver à cet objectif nous allons utiliser la fonction *SetOnClickListener* mais aussi la fonction *if (validate)* qui va nous permettre d'appliquer l'action que nous avons programmer que si les conditions qu'on a établies sont bien rempli.

Ensuite on va définir d'autres actions pour le bouton en ajoutant un autre *SetOnClickListener* dans l'action précédente, celle-ci va nous permettre si la première condition est valide d'appliquer l'action suivante et ainsi de suite. Ici les actions qui nous intéressent sont d'envoyer un mail avec toutes les informations remplies par l'utilisateur dans le formulaire et si cette condition est remplie d'ouvrir votre jeu **bulibox**

PROGRAMME DEFINIR UNE ACTION POUR DES BOUTONS

```
StrButton.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        if (validate())
        { // si les conditions remplis sont bonnes nous allons definir une autre action pour le bouton
start
        StrButton.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                sendMail();

                StrButton.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        open();
                    }
                }
            }
        }
    }
}
```

- Après avoir donner des actions au bouton, nous allons maintenant programmer chaque action. On va d'abord commencer par l'envoi du mail.
- Pour arriver à cet objectif nous allons utiliser la fonction *String* mais aussi la fonction *intent*, *ACTION_SEND*, *EXTRA_EMAIL* et *EXTRA_TEXT*

Tout d'abord, la fonction *ACTION_SEND* qui va nous permettre d'envoyer ce qu'on souhaite.

La fonction *EXTRA_EMAIL* permet de choisir à quel e-mail envoyer les données en utilisant un string et en écrivant l'adresse e-mail de l'agence kiiway.

La fonction *EXTRA_TEXT* permet de définir les données qu'on va envoyer par mail sous forme de texte

PROGRAMME ENVOI DE DONNEES PAR E-MAIL

```
private void sendMail()
{
    String name = UserName.getText().toString();
    String pname = UserLastName.getText().toString();
    String Padresse = UserAdresse.getText().toString();
    String Email = UserEmail.getText().toString();
    String Phone = Userphone.getText().toString();
    String Full = name + " /" + pname + " /" + Padresse + " /" + Email + " /" + Phone + "
";
    Intent emailintent = new Intent(Intent.ACTION_SEND_MULTIPLE);
    emailintent.putExtra(Intent.EXTRA_EMAIL, new String[] {"kiiway01@gmail.com"});
    emailintent.putExtra(Intent.EXTRA_TEXT, Full);

    emailintent.setType("text/rfc822");
    // nous allons definir un message a afficher pour choisir l'application avec laquelle
    on va envoyer l'e-mail
    startActivityForResult(Intent.createChooser(emailintent, "choose an email client"));
}
```

- Après avoir programmer l'envoi du mail, passons maintenant à l'ouverture du jeu **buildbox** à partir de notre portail quand toutes les conditions précédentes sont remplies.
- Pour arriver à cet objectif nous allons utiliser encore une fois la fonction *intent* sauf que pour ouvrir une application externe il va nous falloir utiliser les fonctions *getPackageManager*, *getLaunchIntentForPackage* et enfin *CATEGORY_LAUNCHER*.

Tout d'abord les fonction *getPackageManager* et *getLaunchIntentForPackage* qui vont nous permettre de choisir quelle application ouvrir grâce à son nom de package.

Et ensuite la fonction *CATEGORY_LAUNCHER* qui va nous permettre simplement d'ouvrir n'importe quelle application

PROGRAMME OUVERTURE D'UNE APPLICATION EXTERNE

```
private void open ()  
{  
    Intent i;  
    try {  
        i =  
getPackageManager().getLaunchIntentForPackage("com.companyname.gamename");  
        if (i == null)  
            throw new PackageManager.NameNotFoundException();  
        i.addCategory(Intent.CATEGORY_LAUNCHER);  
        startActivity(i);  
    } catch (PackageManager.NameNotFoundException e) {  
    }  
}
```

INSTALLATION TERMINÉE

- Ouvrez un navigateur web et saisissez l'url :<https://github.com/kiiway/porta>
- **l.git** Ouvrez ensuite votre éditeur de code favori et commencez les modifications

