

Wear OS (Android Wear)



Wear OS
by Google

WearOS

- *Lancé en mars 2014*
- *Version d'Android dédiée aux objets mettables, en particulier aux smartwatches*
- *Peut-être appairé avec un smartphone Android ou Apple*
- *Android Wear 2.0 disponible depuis janvier 2017*
- *Applications autonomes*
- *Navigation repensée*
- *Nouvelles méthodes d'entrées
(Voix, réponses automatiques, Emoji)*

WearOS

- *L'intégration la plus simple avec une application «smartphone» est l'utilisation de notifications*
- *Les notifications sont aussi affichées sur la montre appairée, possibilité de traiter les notifications depuis la montre (actions avec PendingIntent)*

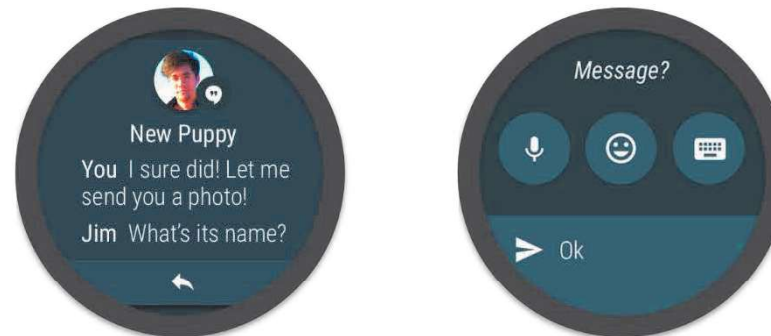


WearOS

- *Lors de la création des notifications, il est possible d'ajouter des actions spécifiques à la montre*

```
.extend(new NotificationCompat.WearableExtender()  
    .addAction(action1)  
    .addAction(action2)  
    .addAction(action3)  
)
```

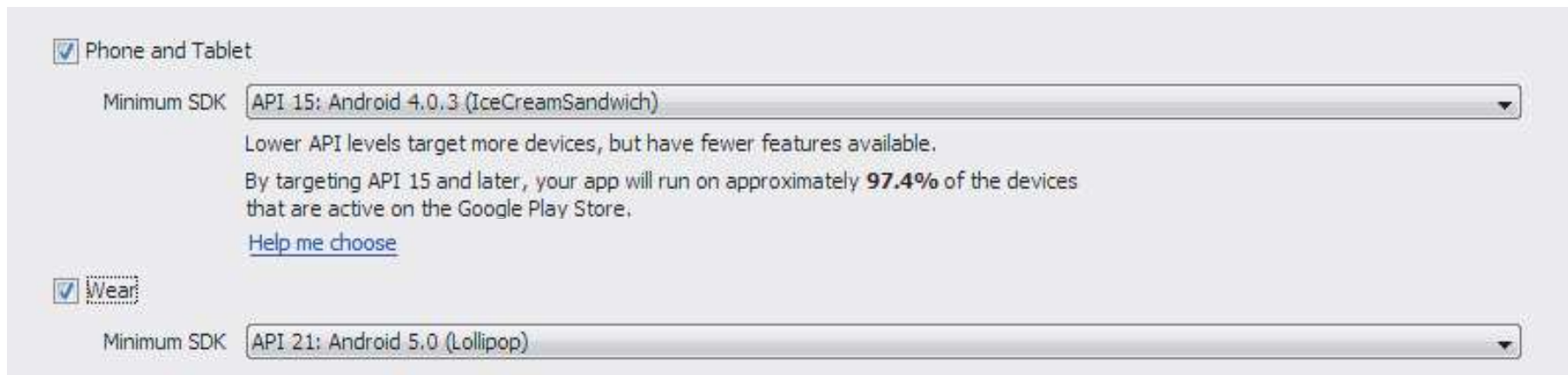
- *Ainsi que de spécifier des options spécifiques aux montres, par exemple une image ou des entrées vocales*



WearOS

- *Wearables App*

Si des fonctionnalités plus avancées sont nécessaires, il faudra se tourner vers la réalisation d'une version dédiée aux wearables de votre application



☒ Phone and Tablet

Minimum SDK API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available.
By targeting API 15 and later, your app will run on approximately **97.4%** of the devices that are active on the Google Play Store.
[Help me choose](#)

☒ Wear

Minimum SDK API 21: Android 5.0 (Lollipop)

WearOS

- *Android Wear 1.0*
 - *Il n'existe pas d'application autonome, on peut réaliser une application «montre» qui accompagne une application smartphone*
 - *La montre communique uniquement avec le téléphone qui sert de passerelle vers l'extérieur*
- *Android Wear 2.0*
 - *Il est possible et même fortement conseillé de développer des applications autonomes pour les montres*
 - *Les montres possèdent une connexion Wifi ou LTE*

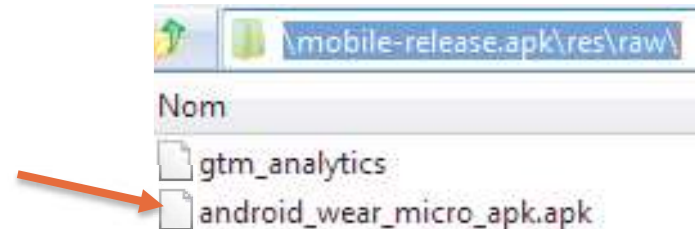
WearOS - Companion app

- *Les deux versions de votre application seront intégrées dans le même projet Android Studio, sous la forme de deux modules*
- *Vous aurez certainement besoin d'un troisième module qui contiendra le code commun aux 2 versions*
- *Lors du développement vous pourrez lancer l'une ou l'autre application*

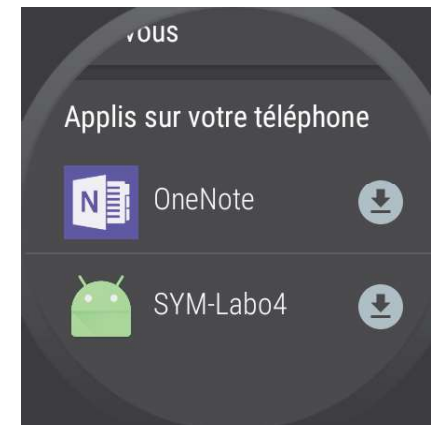


Android Wear - Companion app

- *En mode release, lors de la génération du fichier APK signé pour le smartphone. L'apk de la version wear sera intégré et distribué en même temps:*

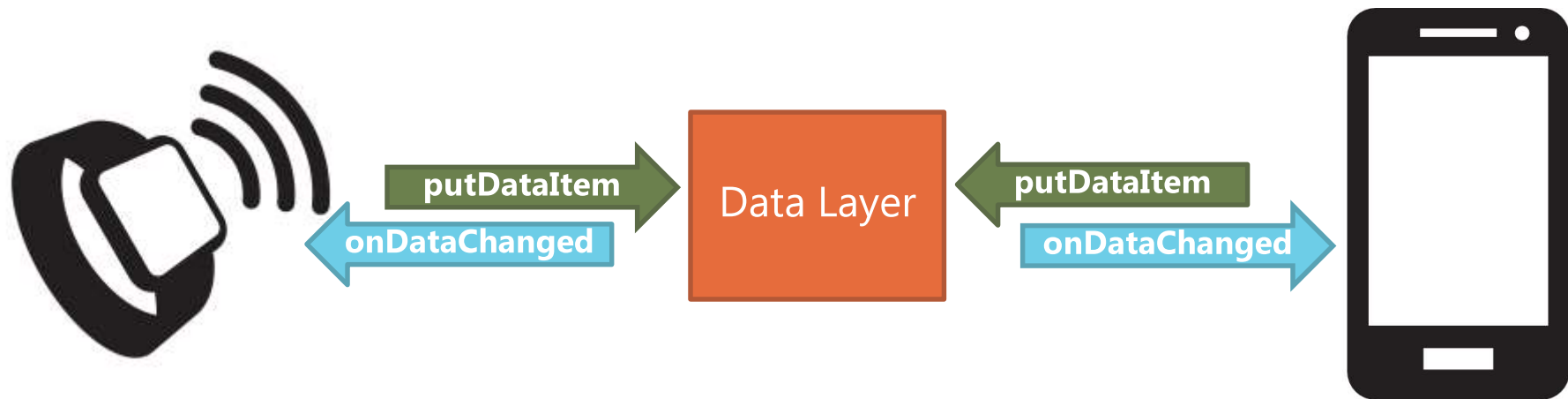


- *Si une montre est appairée avec le téléphone. L'installation de la companion app sera possible depuis l'application Play Store sur la montre (presser ◀ sur l'émulateur)*



WearOS

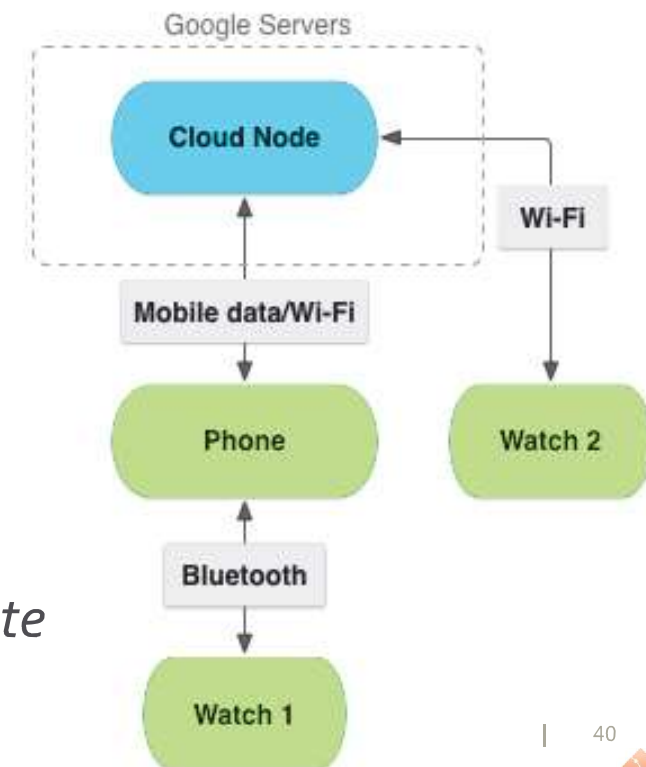
- Les deux applications sont totalement indépendantes l'une de l'autre. Il est toutefois possible d'utiliser la Wearable Data Layer API qui fait partie des play services pour les lier et leur permettre de d'échanger des données*



- Il est aussi possible de transférer des messages ou des assets (images, audios, etc.) entre les 2 applications*

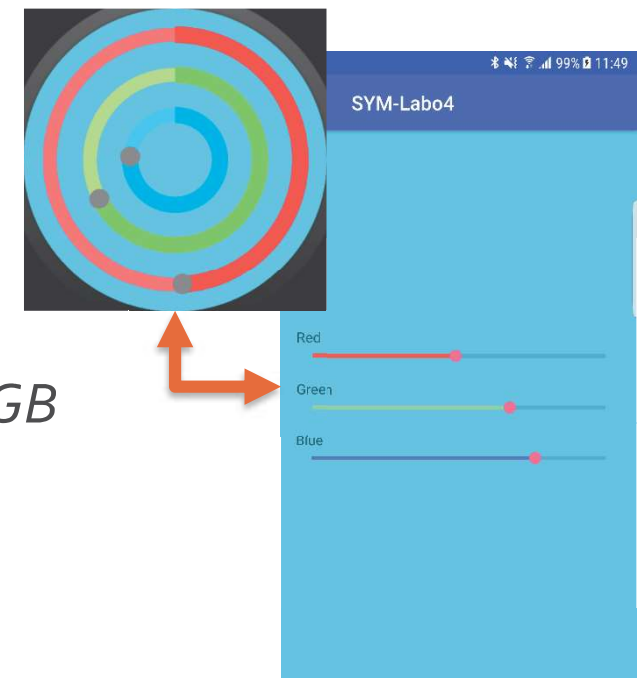
WearOS

- *Une application Android Wear 1.0 ne dispose pas d'un accès direct à Internet*
- *Tout doit passer par l'application smartphone associée
Utilisation de la Data Layer API + Assets*
- *Android Wear 2.0 permet l'accès direct à Internet*
- *Plus besoin d'une connexion bluetooth en permanence avec le smartphone*
- *Toutefois, si le smartphone est à portée les connexions réseaux passeront en priorité via celui-ci (proxy)*
- *Il faut demander explicitement une connexion avec une grande bande passante si on souhaite échanger de larges fichiers*




WearOS - Labo 4

- *Le dernier laboratoire vous permettra de vous familiariser avec Wear OS*
 - *Configuration de l'environnement de développement*
- *Emulateur de smartwatch connecté à un smartphone physique*
- *Une manipulation avec les notifications et les PendingIntent (actions) associés*
- *Ainsi qu'une manipulation avec la Data Layer API entre une application «montre» et une application «smartphone»*
 - *3 sliders permettant de définir une couleur RGB*
 - *l'écran du smartphone est mis à jour avec la couleur correspondante*
 - *Et vice-versa*



WearOS - Labo 4

- *La configuration de l'environnement de développement n'est pas forcément intuitive, voici quelques informations:*
 - *Il faut absolument disposer d'un smartphone physique*
 - *Possibilité d'utiliser un émulateur de montre*
- *Le smartphone doit posséder l'application Wear OS qui permet d'appairer le téléphone et une montre (ou un émulateur)*
- *Comme nous allons utiliser les play services, vous devrez obligatoirement utiliser l'image la plus récente de l'émulateur (9.0)*
Il faudra aussi mettre à jour les applications préinstallées sur
 *l'image, soit associer votre compte Google pour que le Play Store soit fonctionnel*

WearOS - Labo 4

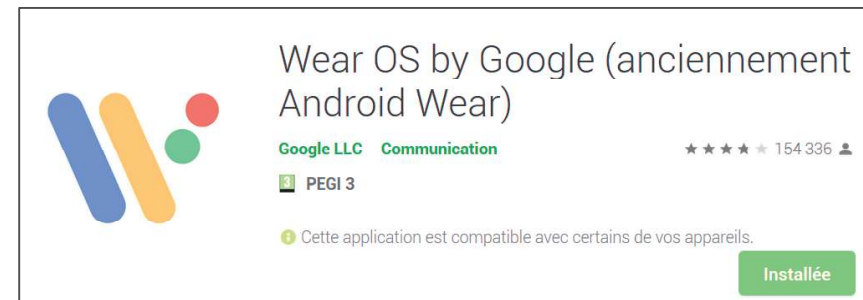
- La commande ***adb devices*** vous permet de lister les devices connectés:

```
PS C:\Users\fabien.dutoit> adb devices
List of devices attached
emulator-5554    device
CB5A28DUJ9      device
```

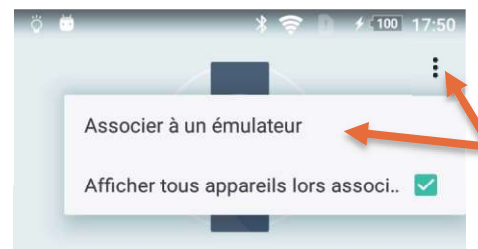
- La commande ***adb -d forward tcp:5601 tcp:5601*** va permettre au device de communiquer avec l'émulateur, elle devra être répétée en cas de déconnexion du smartphone
- L'utilitaire ***adb*** se situe dans ***<android_sdk_path>/platform-tools*** vous pouvez l'ajouter à votre variable d'environnement ***PATH***

Wear OS - Labo 4

- Vous devrez installer l'application Wear OS sur le smartphone*



- Cette application permettra d'appairer votre montre et votre smartphone. Dans le cas de l'émulateur la détection automatique ne fonctionne pas, vous devrez forcer l'association de l'émulateur*



Connecter votre montre

Appuyez sur le nom de la montre lorsqu'il s'affiche dans la liste ci-dessous.

WearOS - Labo 4

Manipulations

- *Notifications*

Dans cette première manipulation nous souhaitons afficher des notifications sur la montre. Lorsque l'on clique sur la notification ou une action depuis la montre, un PendingIntent est envoyé au téléphone qui affichera un Toast.

3 types de notification à réaliser:

- *Notification simple*
- *Notification avec des Action Buttons*
- *Notification avec des actions spécifiques à la montre*

WearOS - Labo 4

Manipulations

- *Application dédiée*

Le but de cette manipulation est de faire communiquer une application dédiée (\neq autonome) sur une montre connectée avec l'application correspondante sur le smartphone à l'aide de la Wearable Data Layer API

- *Sur la montre nous vous fournissons une activité avec 3 sliders circulaires qui permettent de sélectionner 3 composantes couleurs (RGB)*
- *Cette couleur est envoyée à l'application «compagnon» sur le téléphone qui affichera cette couleur en fond d'écran*
- *Et vice-versa*
- *Possibilité de tester votre code sur une vraie montre*