



TypeScript



# INF 3511 Programmation des Mobiles



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique



# A propos de moi

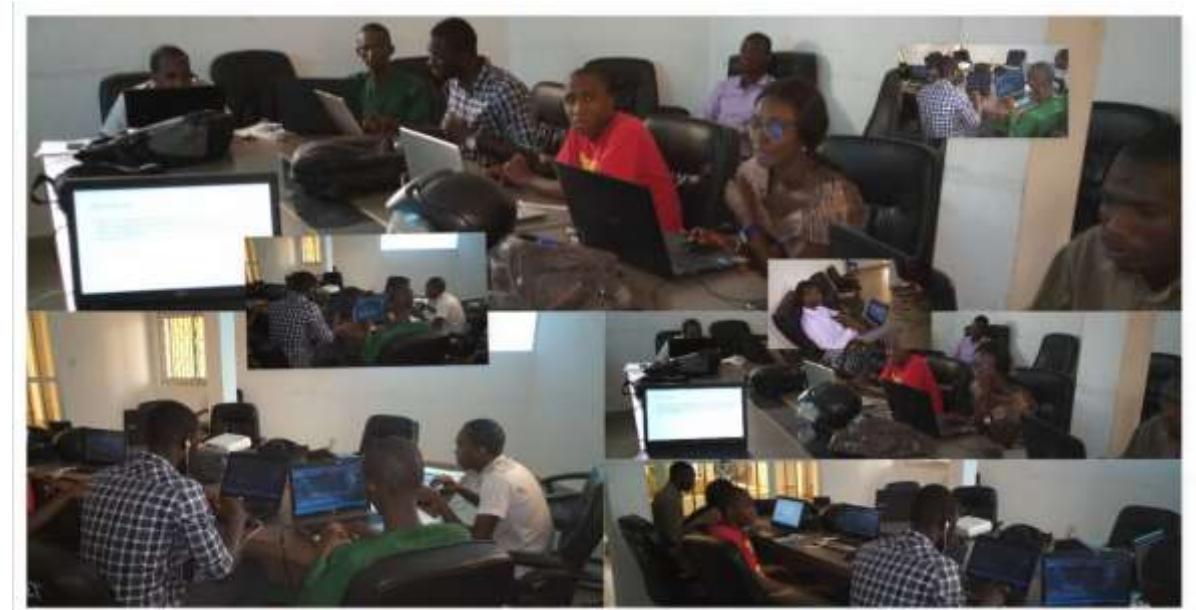


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
3. Approches de développement mobile
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android



# Description de l'ECUE

- Ce cours s'adresse aux étudiants de Licence Informatique Option Génie Logiciel, il porte sur :
  - les méthodes de développement d'applications mobiles;
  - le développement d'applications mobiles hybrides et natives de qualité (fiables, faciles à utiliser, à comprendre et à modifier).
- Ce cours est principalement basé sur:
  - Supports disponibles sur le web

# Objectifs/Compétences visé(e)s

- **Objectif général** : l'objectif du cours est de maîtriser les concepts, enjeux et les techniques de base de la programmation d'applications mobiles.
- **A la fin de ce cours, l'étudiant devra être capable de :**
  - Mettre en œuvre les méthodologies pour la conception et le développement d'applications mobiles ;
  - Développer et déployer des applications mobiles qui s'exécutent à la fois sur les appareils iOS et Android, Windows Phone.
  - Comparer les défis du développement mobile natif et cross-plateforme; Utiliser Ionic pour créer des applications mobiles hybrides et Android pour des applications natives;
  - Créez des vues complexes à l'aide de mises en page et de contrôles avancés; Créer des listes scrollables; Contrôler la navigation dans les applications;
  - Lier les données aux formulaires et enregistrer les données dans les bases de données ;
  - Déployer des applications dans l'App Store et dans Google Play Store.

# Evaluation du cours

- Les apprentissages sont évalués par un examen pratique (projet) individuels à rendre à la fin du cours.
- Le projet consistera au développement d'une solution mobile Cross-Platform mettant en œuvre les concepts, normes et méthodologies vus dans le cours relevant des couches présentation, métier et données.
- Ce projet à rendre devra faire l'objet d'un petit rapport dont le gabarit sera donné par clé usb et une séance de présentation diapositive.

# Prérequis

- Langage de programmation orienté objet comme C# ou Java et une connaissance de base du JavaScript
- Expérience avec Visual Studio ou un autre EDI
- Connaissance du HTML et des styles CSS
- Dans la mesure du possible une machine avec au moins **08 Go de RAM** et une connexion internet avec un bon débit



# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.



TypeScript



# INF 3511 Programmation des Mobiles: Environnements Mobiles



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique



Partie 1 :

Généralités sur les Technologies Mobiles



# A propos de moi

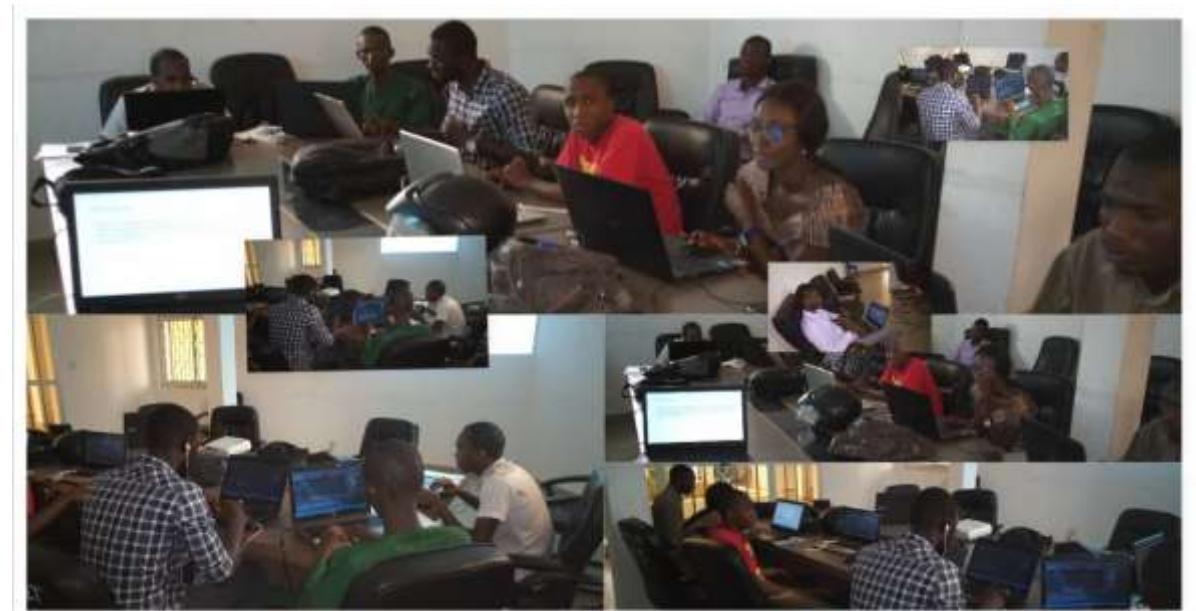


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

- 1. Généralités sur les Technologies Mobiles**
2. Périphériques et Systèmes d'exploitations mobiles
3. Approches de développement mobile
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android



# Introduction

- Les **applications mobiles** sont des applications logicielles conçues pour fonctionner sur les smartphones, tablettes et autres appareils mobiles.
- Elles sont disponibles par le biais des plateformes de distribution natives. Typiquement, dans des App Stores (magasins d'applications) qui sont exploités par les propriétaires du système d'exploitation mobile.
- Certains des magasins les plus populaires sont l'App Store d'Apple, Google Play, ainsi que Windows Phone Store et BlackBerry App World.



# Introduction



- Quatre approches de développement Mobile: **Native, Web, Hybride, Cross-Platform**. L'approche **cross plate-forme** permet d'écrire une logique métier et de créer une interface utilisateur dans un cadre commun qui construit des applications pour iOS, Android et éventuellement Windows. *Voir écrire une fois, exécuter n'importe où...*



# Introduction



- En 2018, en un an, 194 milliards de téléchargements ont été effectués dans le monde, soit plus de 530 000 par jour.
- Les revenus générés par ces services en tout genre (jeux, messageries, outils, sites d'information, etc.) ont rapporté à leurs éditeurs, sur l'année 2018, à travers les « stores » de Google et d'Apple, 101 milliards de dollars (57 640 milliards de francs CFA près de 15 fois le budget 2019 de l'état du Sénégal).

Source: <https://www.clubic.com/application-mobile/actualite-849849-100-dollars-achats-applications-mobiles-2018.html>

# Panorama de l'écosystème Mobile en 2016

- L'écosystème mobile se développe à grande vitesse pour devenir logiquement de plus en plus complexe à suivre avec une multitude d'intervenants, de prestataires, supports, outils analytiques... qui représentent plus de 350 sociétés, 28 Milliards \$ d'investissement, le tout pour une valorisation estimée à 405 Milliards \$ en 2016.

<https://programmatique-marketing.fr/2016/04/30/panorama-de-lecosysteme-mobile-en-2016/>



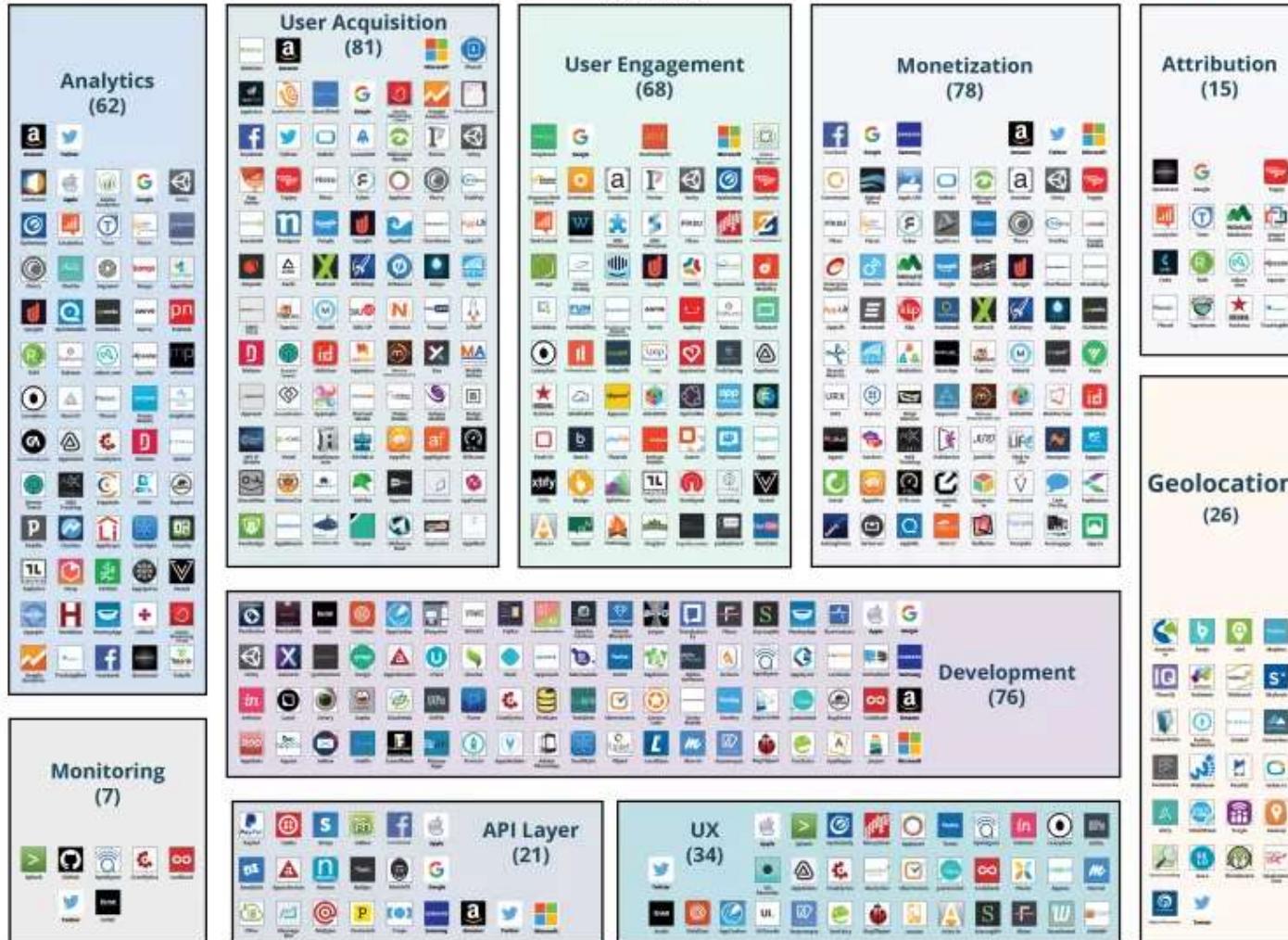
# Panorama de l'écosystème Mobile en 2016

## Mobile Foundations Landscape

DESIGNED BY  
JOHN KOETSIER

March 21, 2016

POWERED BY  

# Travaux dirigés

- Qu'est-ce qu'une application mobile ?
- Où sont stockées les applications mobiles ?
- Citer quelques acteurs de l'écosystème mobile
- En vous référant à la diapositive n°18 lister les domaines d'intervention des acteurs de l'écosystème et pour chaque domaine lister au moins trois acteurs
- Quels sont les enjeux financiers liés au développement mobile ?
- Lire l'article <https://programmatique-marketing.fr/2016/04/30/panorama-de-lecosysteme-mobile-en-2016/> et proposer un résumé reprenant les parties pertinentes pour dresser le panorama en 2016 de l'écosystème mobile.

# L'omniprésence des technologies mobiles



# Chiffres-clés

- 3,8 milliards d'utilisateurs mobiles dans le monde en 2008 qui ont générés 742 milliards de dollars de revenus. D'après la [GSM Association \(GSMA\)](#), qui représente 850 opérateurs de téléphonie mobile à travers 218 pays du monde, il y aurait désormais plus de 5 milliards d'abonnés mobiles de par le monde...
- 366 milliards de chiffre d'affaires en 2011, selon l'ARTP au Sénégal pour 16,1 millions d'abonnés (88%), au 30 juin 2018. **Un chiffre d'affaires de 506,4 milliards de FCFA sur la première moitié de l'exercice 2018(Août), en hausse de 22,8 Mds par rapport aux résultats du 1er semestre de 2017 pour la SONATEL en 2018 pour un parc actuel de téléphonie mobile de plus de 16.141.304 lignes.**
- Au Sénégal l'**Internet mobile**, hors clés, représente 8.483.435 d'utilisateurs.

# Chiffres-clés

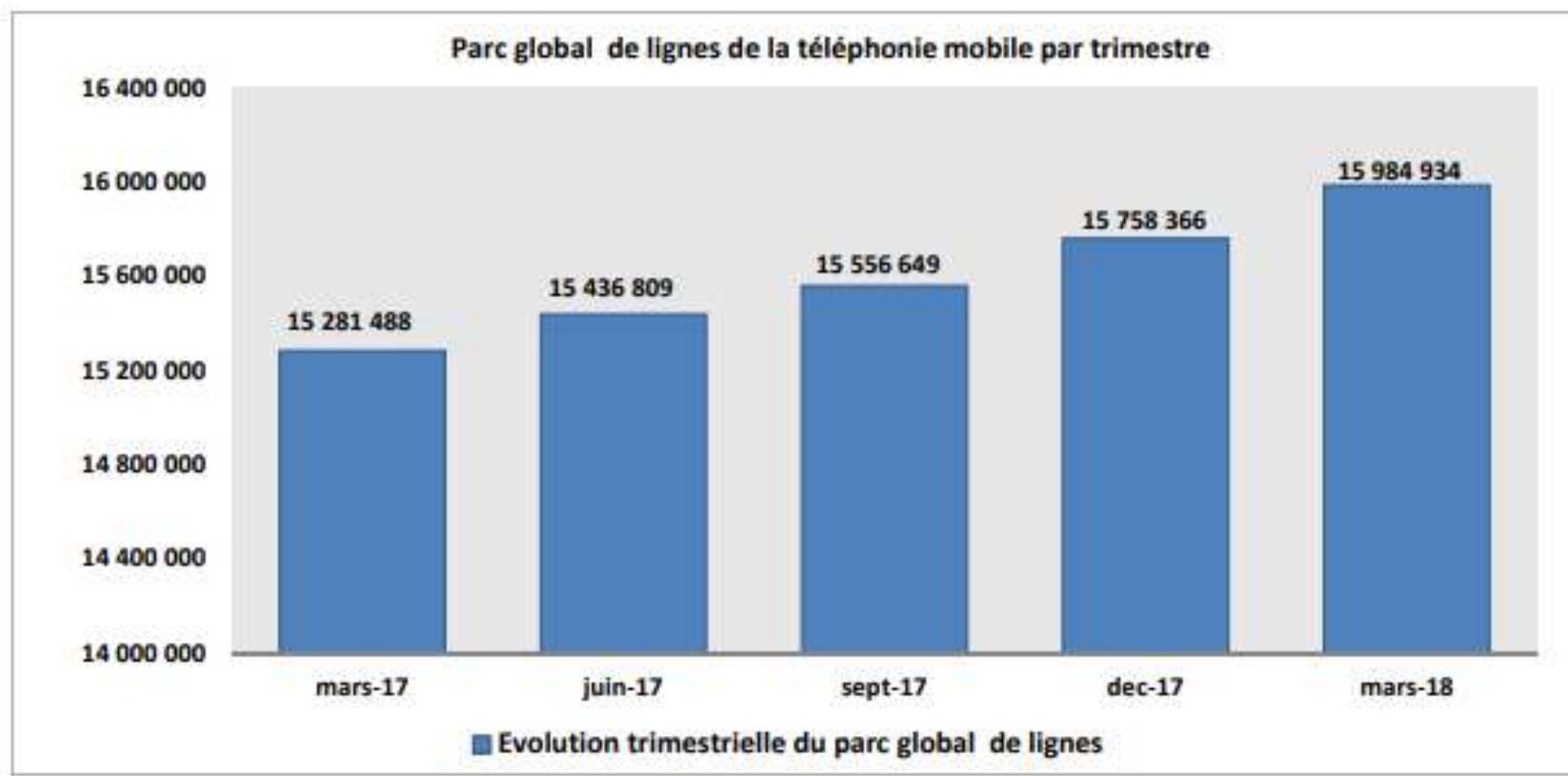
- Fin 2017, l'Afrique de l'Ouest, définie par la région géographique correspondant aux 15 États membres de la Communauté économique des États de l'Afrique de l'Ouest (CEDEAO), comptait 176 millions d'abonnés uniques
- D'ici 2025, l'Afrique de l'Ouest comptera environ 72 millions de nouveaux abonnés mobiles

<https://www.gsmaintelligence.com/research/?file=dd7760bf439236e808ea61ee986845eb&download>

# Forte évolution du parc de téléphonie mobile

[https://www.artpsenegal.net/sites/default/files/docs\\_actualites/resume\\_executif\\_du\\_rapport\\_t2\\_2018.pdf](https://www.artpsenegal.net/sites/default/files/docs_actualites/resume_executif_du_rapport_t2_2018.pdf)

Le parc de lignes de téléphonie mobile s'élève à 15.984.934 lignes au 31 mars 2018, soit une hausse de 1,43% par rapport au trimestre précédent.

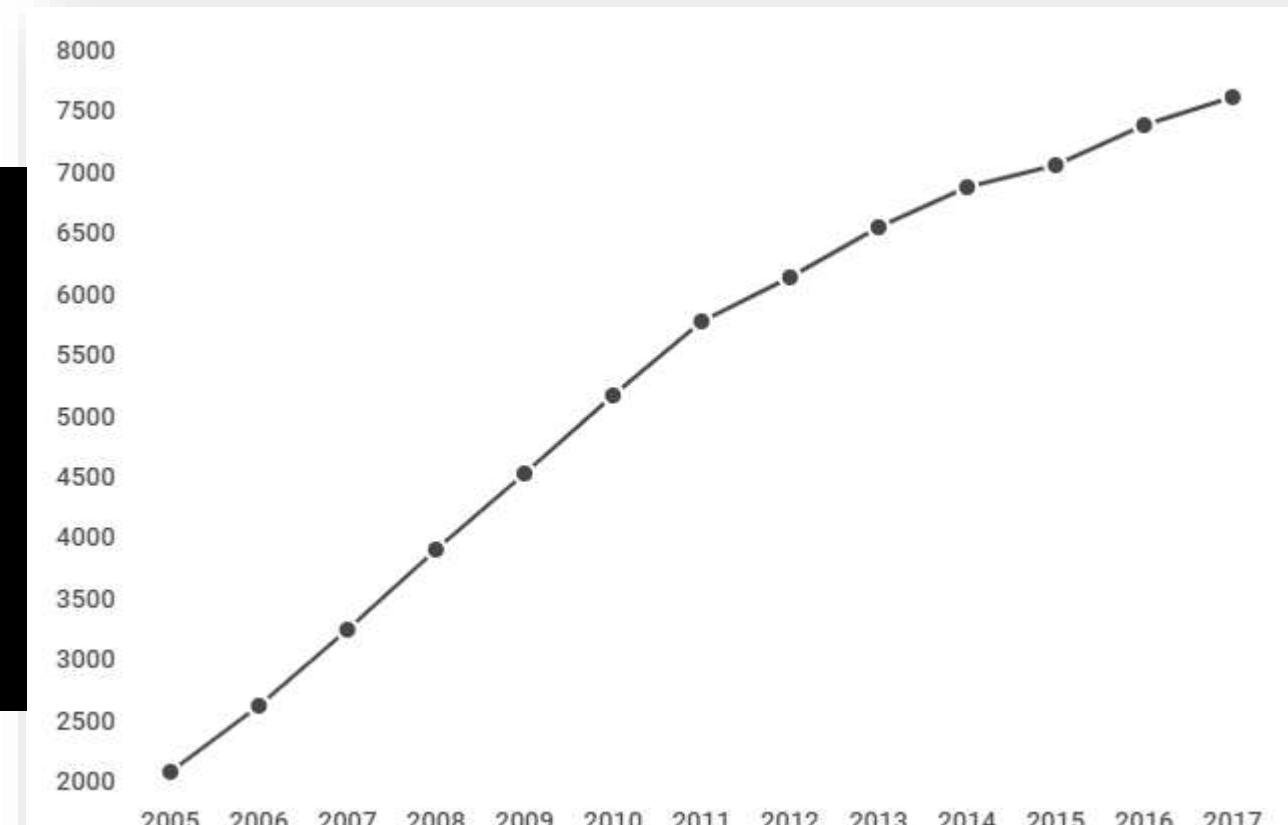


Situation au Sénégal

# Le nombre d'abonnés au téléphone mobile dans le monde(en millions)

<https://www.journaldunet.com/ebusiness/internet-mobile/1009553-monde-le-nombre-d-abonnes-au-telephone-mobile/>

Dans le monde, ce sont près de **7,7 milliards d'abonnements mobiles** qui étaient souscrits fin 2017, soit plus de la totalité de la population mondiale, selon les estimations de l'International Telecommunication Union. Cela correspond ainsi à un **taux de pénétration de 103,5%**. 6,1 milliards de ces abonnements ont été souscrits dans des pays en développement.



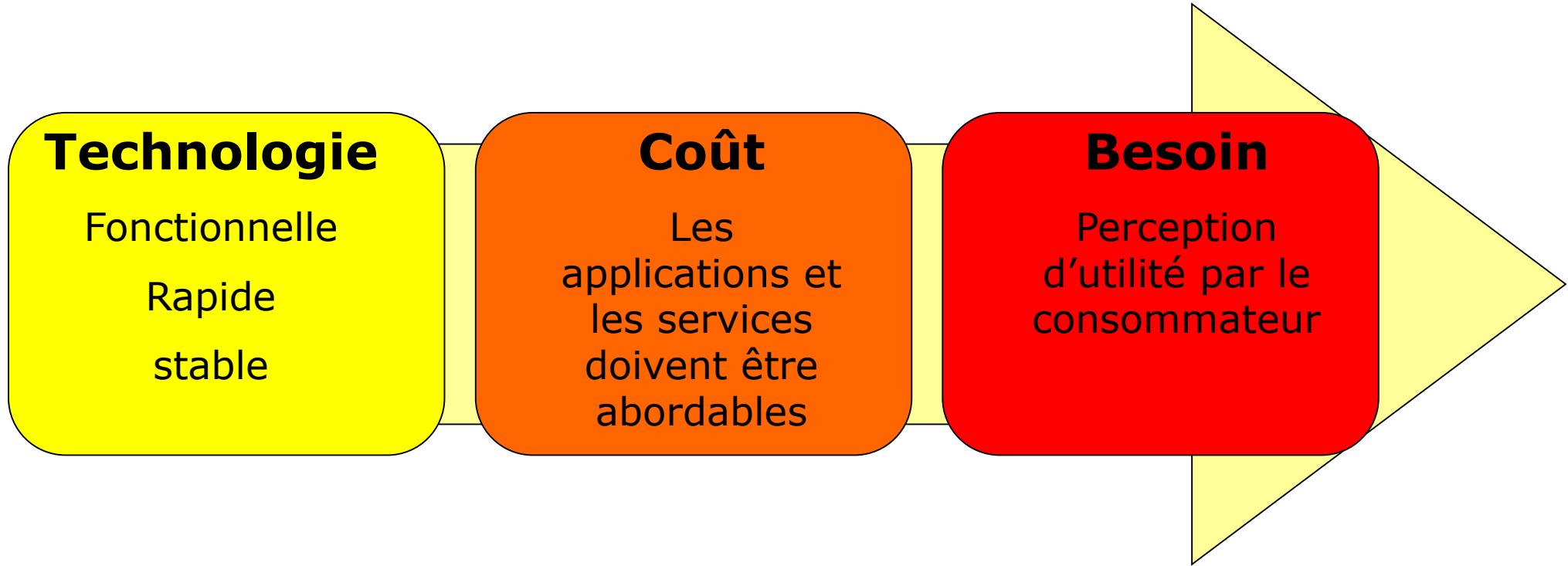
# Les 7 avantages du média mobile

- C'est le premier « Personal Mass Media »,
- Il est porté en permanence par son détenteur,
- Il est toujours connecté ou connectable,
- Il peut embarquer un système de paiement,
- Il constitue « un point d'inspiration » (photo, vidéo, blogging...)
- Il est plus pertinent en termes de mesures d'audience que le Web,
- Il est socialement contextuel.

**« The Mobile is not the Widest Reaching Media,  
but the Most Intensely Involving Media »  
(Tomi Ahonen)**



# Les 3 ingrédients du succès pour l'industrie mobile



# Des mobinautes exigeants

- « Always On »...
  - L'ordinateur personnel n'est plus l'outil par défaut de connexion à l'Internet,
  - Le consommateur attend des offreurs de services que leurs contenus soient disponibles n'importe où, n'importe quand (« Everyone is in the Shop »),
  - Ce sentiment s'accompagne d'un besoin d'instantanéité (éviter les frustrations),
  - Csq: Ces réalités conduisent à des comportements de surf mobile totalement différents de ce qu'on connaissait auparavant!



Prof. Ousmane SALL, Univ. Thiès, SN



Programmation Applications Mobiles



27

# Quelques chiffres...

- Le mobile est la première interface avec laquelle on interagit avant de s'endormir....et la première avec laquelle on se réveille!
- **51,3 % des utilisateurs en France accèdent à internet via un mobile ou une tablette contre 48,7% via un ordinateur.** Depuis octobre 2016, l'accès à internet via mobile et tablette a dépassé l'accès à internet via l'ordinateur



# Et le Sénégal dans tout cela ?

**TELECOM** **Sénégal : Plus De 15 Millions De Lignes De Téléphonie Mobile Recensées Au Premier Trimestre 2017**



Le parc de lignes de téléphonie mobile recensé au Sénégal durant le premier trimestre 2017 s'est élevé à 15.281.488, a appris mardi APA auprès de...

[http://www.seneweb.com/news/Societe/senegal-plus-de-15-millions-de-lignes-de\\_n\\_221409.html](http://www.seneweb.com/news/Societe/senegal-plus-de-15-millions-de-lignes-de_n_221409.html)

# Et le Sénégal dans tout cela ?



# Au Sénégal ?



# Travaux dirigés

- Qu'est-ce que GSMA ?
- Quel est le nombre d'abonnés mobiles de par le monde ?
- Donner la taille du parc actuel de téléphonie mobile au Sénégal ?
- Quel est la part de la population mondiale possède un téléphone mobile ?
- Expliquer en quelques mots les avantages liés au mobile
- Quels sont les causes de succès du mobile ?
- Lister 05 plateformes mobiles existantes
- Citer deux plateformes dont la commercialisation a connu un échec
- Quelle nouvelle maladie est associée au mobile ?
- Décrire en quelques mots la situation au Sénégal
- Proposer un résumé du rapport 2018 de l'ARTP disponible en bibliographie



TypeScript



# INF 3511 Programmation des Mobiles: Environnements Mobiles



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

Partie 2 :

Périphériques et Systèmes d'exploitations  
mobiles



# Périphériques et Systèmes d'exploitations mobiles

# A propos de moi

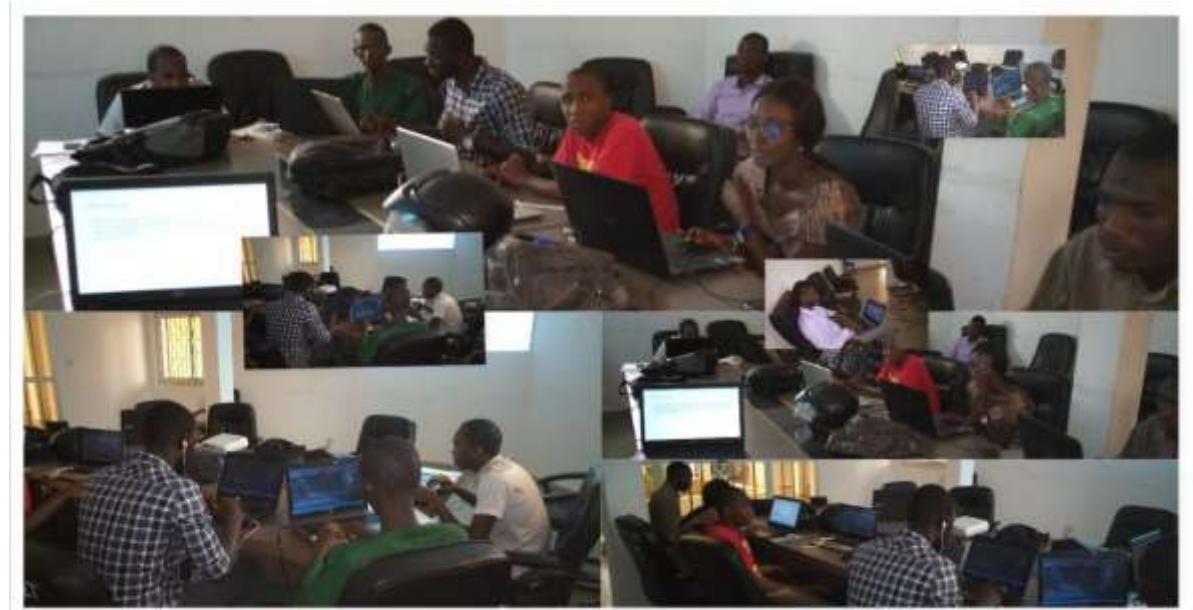


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. **Périphériques et Systèmes d'exploitations mobiles**
3. Approches de développement mobile
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android

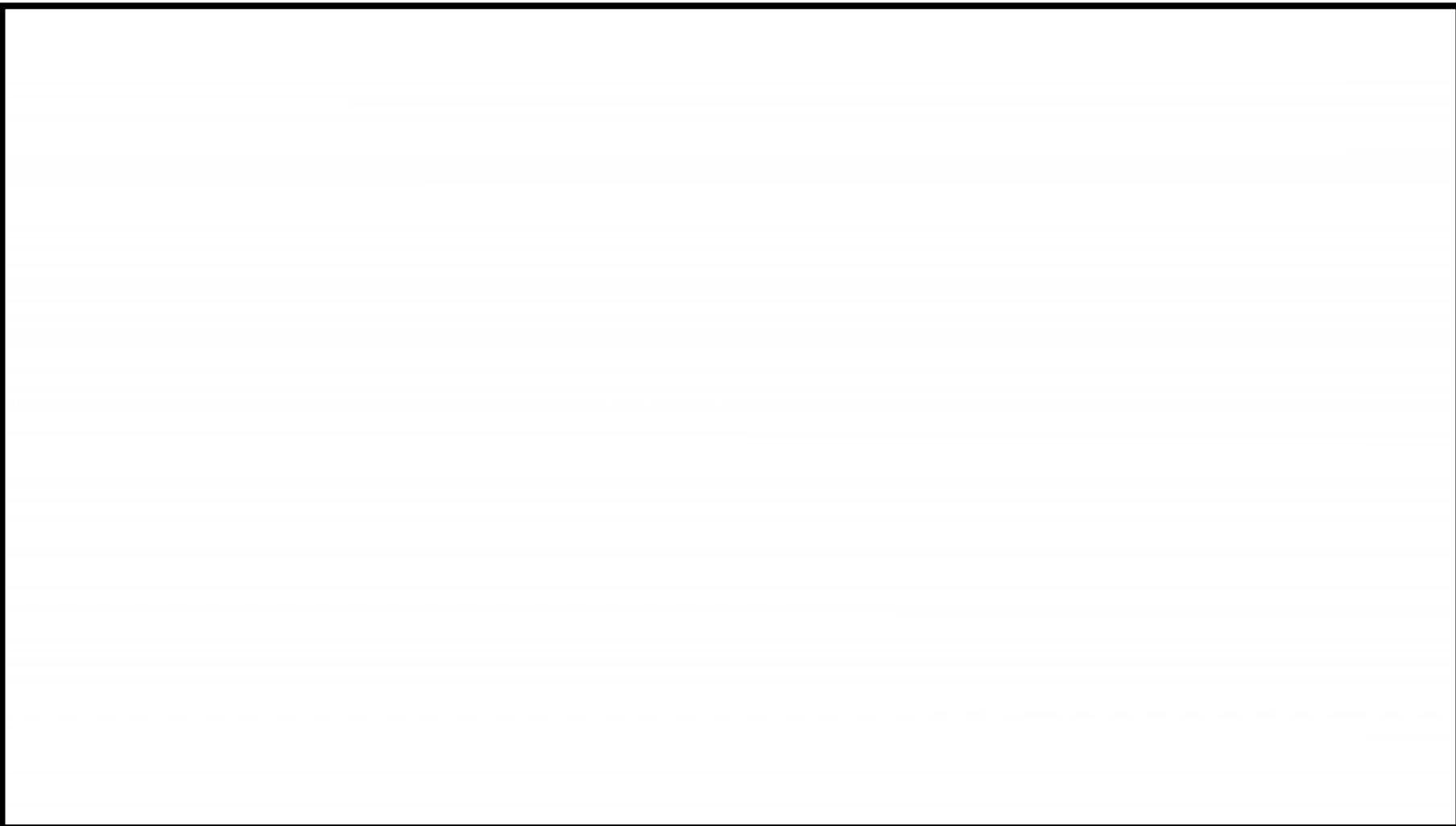


# Périphérique mobile

- Un **appareil mobile** (traduction littérale du terme anglophone « *mobile device* ») est un appareil informatique portatif utilisable de manière autonome lors d'un déplacement.
- Les *appareils mobiles* sont de petites tailles — certains peuvent être mis dans les poches. Ils sont typiquement dérivés des téléphones mobiles, et permettent d'accéder au Web, de lire du courrier électronique, de prendre des photos, de jouer à des jeux vidéo, d'écouter de la musique, de regarder des clips vidéo ou bien de télécharger des applications. Ils peuvent également comporter un calendrier ou un carnet d'adresses.

[https://fr.wikipedia.org/wiki/Appareil\\_mobile](https://fr.wikipedia.org/wiki/Appareil_mobile)

# Evolution des périphériques mobiles



# Evolution des périphériques mobiles

- L'historique des téléphones mobiles commence avec Martin Cooper commence le 3 Avril 1973, un ingénieur senior travaillant pour Motorola et qui s'appelle [Martin Cooper](#) a utilisé un téléphone portable pour appeler un concurrent potentiel sur le marché de la téléphonie mobile. Ce fut le premier coup de téléphone mobile de l'histoire.
- Le téléphone de Cooper avait les dimensions suivantes: 228,6mm par 127mm par 44,4mm et pèse 1,1 kg. C'était un prototype qui a pris environ dix heures pour se charger et avait une autonomie de 30 minutes.



<https://www.stelladoradus.fr/lhistorique-des-telephones-mobiles/>

10 ans plus tard, en 1983, le Motorola **DynaTAC 8000X** est venu sur le marché et coûte la somme astronomique de **3.000€**. C'était le premier téléphone mobile offrant **30 minutes de conversation**, avec une installation pour stocker **30 numéros** et une période en veille de 6 heures.

# Evolution des périphériques mobiles

Les technologies mobiles ou de téléphonie mobiles permettent une communication sans fil (téléphonique, Internet, ..) qui fonctionne même lorsque l'utilisateur est en mouvement.

- 1980-1990



A) Mobile Phones 1985-86: Motorola 8000S, Mitsubishi Roamer, Technophone PC135



A) Mobile Phones 1987-1988: NEC 9A, Motorola 8500X, Nokia Cityman 1320, Philips PRC30E, British Telecom Coral

Même si ça paraît incroyable aujourd'hui, ces appareils étaient révolutionnaires à l'époque car ils permettaient de téléphoner sans être obligé d'être à côté du combiné.

# Evolution des périphériques mobiles

Les premiers appareils de téléphonie mobiles de la décennie 1990 adressaient la question de la taille des appareils et de l'autonomie suffisante pour pouvoir émettre des appels durant toute la journée sans le recharger. Coté fonctionnalité c'était très basique avec seulement la possibilité d'enregistrer quelques numéros de téléphones. Mais le plus important est que cette décennie a vu le développement fulgurant de GSM qui signe la fin de l'ère analogique.



**GSM phones 1993-4: Ericsson GH337, Motorola 5200, One2One M200, Nokia 2140**



**GSM phones 1995-6: Nokia 1610, Nokia 2110i, Nokia 8110, Motorola 7500, Ericsson GA318, Motorola StarTAC**



**GSM phones 1999: Motorola Timeport L7089, Ericsson T28, Nokia 8210, Motorola v3688, Nokia 3210, Sagem 815**

# Evolution des périphériques mobiles

- **Depuis 2000**

- Des années 2000 à nous jours nous sommes dans la période de la convergence technologique et des objets connectés (diversification des équipements et des services).
- Cette révolution est arrivée avec l'iPhone d'Apple, puis celle du système d'exploitation Android, enfin la création d'une ensemble d'équipements connectés comme les tablettes, les montres connectés, la télévision, ...



# Smartphone

- Un smartphone est un ordinateur de poche qui combine les fonctions d'un téléphone mobile avec celles d'un ordinateur (accéder au web, envoyer des e-mails, et écouter de la musique).
- BlackBerry a été un des premiers appareils de ce type, lancé par le groupe canadien Research In Motion(RIM) en 1999.



# Smartphone

- L'arrivée des smartphones a changé les habitudes d'utilisation. Outre les appels et les messages SMS, les smartphones sont couramment utilisés pour:
  - accéder à Internet,
  - envoyer des e-mails,
  - prendre des photos,
  - consulter les prévisions météo,
  - jouer,
  - ou accéder aux réseaux sociaux.



# Formes variées, tailles, capacités



3

# Système d'exploitation mobile

- Un **système d'exploitation mobile**, appelé aussi un **système mobile**, est un logiciel qui gère un appareil mobile intelligent tel qu'une tablette ou un smartphone.
- Les systèmes d'exploitation mobiles modernes combinent les caractéristiques d'un **système d'exploitation** d'un ordinateur personnel avec d'autres fonctionnalités comme un écran cellulaire, ...

Un **système d'exploitation mobile** est un **système d'exploitation** conçu pour fonctionner sur un appareil **mobile**. Ce type de **système d'exploitation** se concentre entre autres sur la gestion de la connectivité sans fil et celle des différents types d'interface.

Système d'exploitation mobile — Wikipédia  
[https://fr.wikipedia.org/wiki/Système\\_d%27exploitation\\_mobile](https://fr.wikipedia.org/wiki/Système_d%27exploitation_mobile)

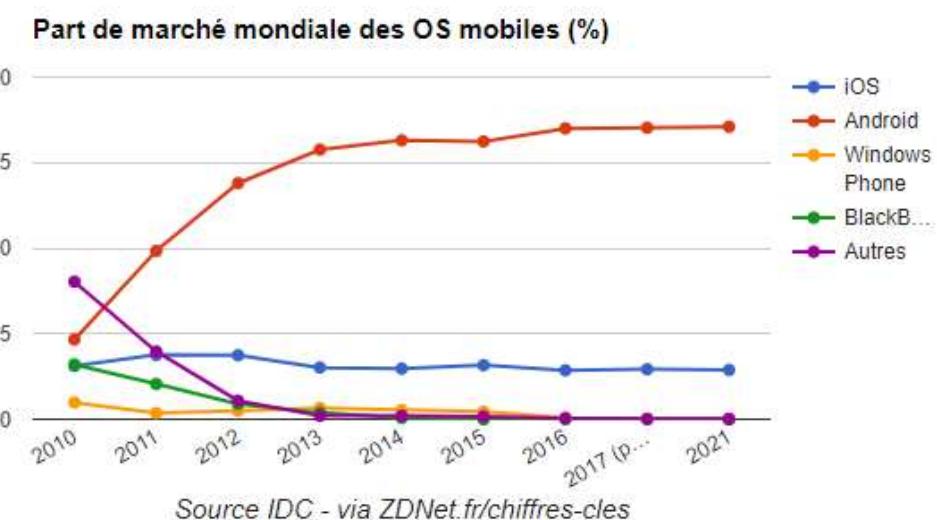


# Part de marché mondial des OS mobiles

Quarter	2016Q4	2017Q1	2017Q2	2017Q3	2017Q4	2018Q1	2018Q2	2018Q3
<b>Android</b>	81,4%	85,0%	88,0%	87,6%	80,3%	84,3%	87,8%	86,8%
<b>iOS</b>	18,2%	14,7%	11,8%	12,4%	19,6%	15,7%	12,1%	13,2%
<b>Others</b>	0,4%	0,2%	0,2%	0,1%	0,1%	0,0%	0,1%	0,0%
<b>TOTAL</b>	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%

<http://www.idc.com/prodserv/smartphone-os-market-share.jsp>

- Android et iOS règnent
- Android sur près de 9 smartphones sur 10
- Windows a fait naufrage



<http://www.zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.htm>

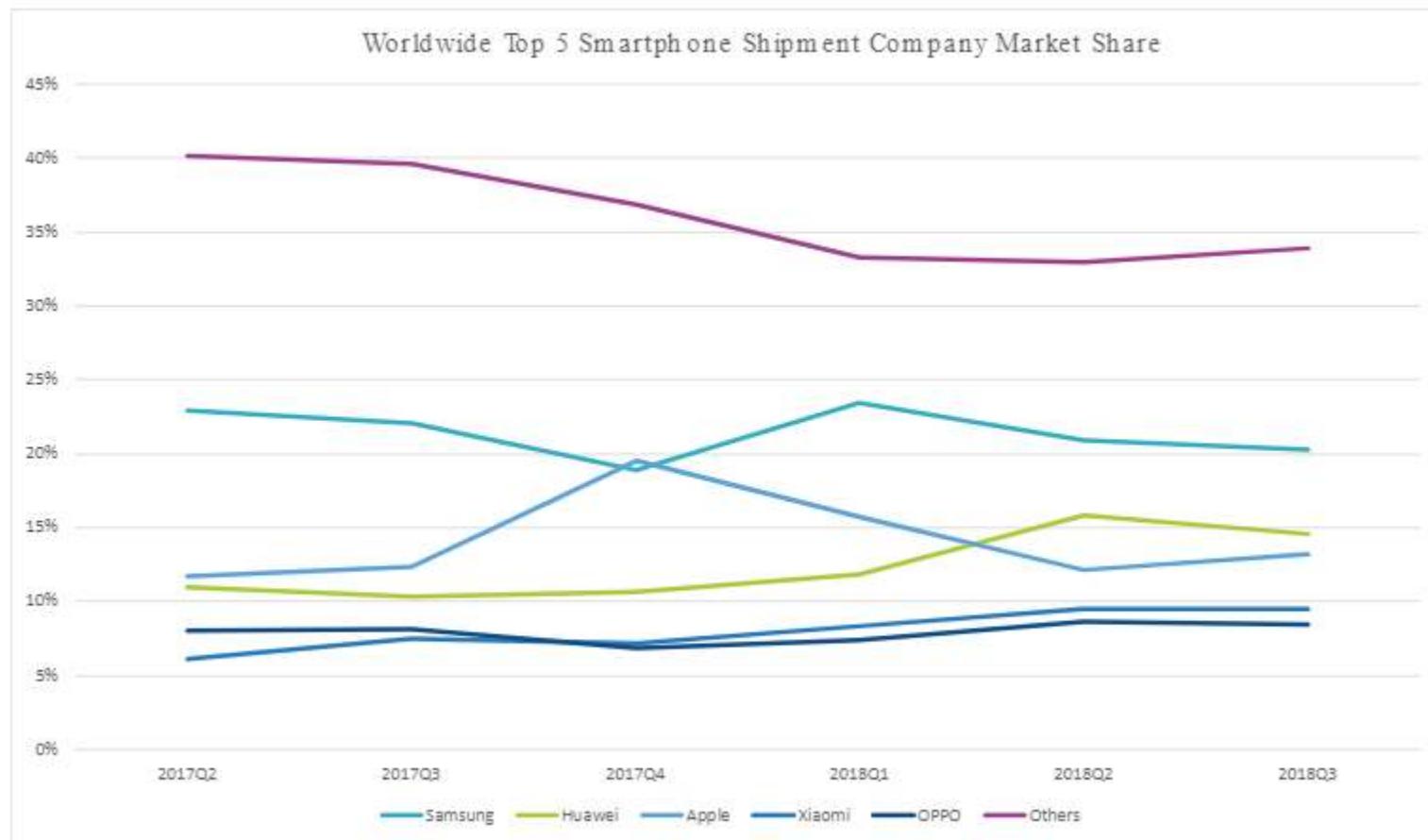
Pour développer des applications mobiles, il est nécessaire de comprendre les principes sous-jacents de plates-formes. Ces plates-formes mobiles offrent l'OS de base un environnement de développement et un market pour publier ces applications. Les principales plateformes mobiles sont : Android, iOS, Windows Phone, BlackBerry OS, ...

# Part de marché mondial selon les marques

Quarter	2017Q1	2017Q2	2017Q3	2017Q4	2018Q1	2018Q2	2018Q3
<b>Samsung</b>	23,2%	22,9%	22,1%	18,9%	23,5%	21,0%	20,3%
<b>Huawei</b>	10,0%	11,0%	10,4%	10,7%	11,8%	15,9%	14,6%
<b>Apple</b>	14,7%	11,8%	12,4%	19,6%	15,7%	12,1%	13,2%
<b>Xiaomi</b>	4,3%	6,2%	7,5%	7,1%	8,4%	9,5%	9,5%
<b>OPPO</b>	7,5%	8,0%	8,1%	6,9%	7,4%	8,6%	8,4%
<b>Others</b>	40,2%	40,1%	39,6%	36,8%	33,2%	32,9%	33,9%
<b>TOTAL</b>	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%

<https://www.idc.com/promo/smartphone-market-share/vendor>

# Part de marché mondial selon les marques



# Choix de la plate-forme mobile

- La première étape dans le développement d'applications mobiles est de décider quelle plate-forme pour développer.
- Les paramètres suivants vous permettent de choisir celle qui est la plus adaptée:
  - **Le public cible:** En fonction du niveau de diffusion de votre application, vous pouvez choisir la plateforme dominante sur le marché;
  - **Marché:** Vous pouvez choisir votre application en comparant les parts de marchés de l'application.

# Choix de la plate-forme mobile

- **Caractéristiques spécifiques:** Chaque plateforme a des caractéristiques spécifiques qui peuvent être utilisées pour votre application. Considérer les caractéristiques des dispositifs mobiles lors de la conception de votre application peut permettre de gagner du temps de développement, d'avoir une meilleure sécurité, ...

Mobile Plateform	Editeur	Description
Android	Google	Android est un système d'exploitation développé par Google Inc. et publié en Septembre 2008, suivie par le lancement d'Android Market en Octobre 2008 avec des applications qui sont disponibles fournisseur libre de logiciels open source pour tout appareil intelligent.
iOS	Apple	iOS est un système d'exploitation mobile développé par Apple Inc., d'abord été dévoilée en 2007 pour l'iPhone d'Apple, et distribué exclusivement au matériel Apple. Il a la deuxième plus grande base installée de smartphones. Il est fermé et la source exclusive, construite sur l'open source Darwin OS noyau.
Windows Phone	MicroSoft	Windows Phone est édité par Microsoft, a été lancé en Octobre 2010 avec sa propre boutique d'applications. Il est développé avec le code fermé et propriétaire. Il a la troisième plus grande base installée de smartphones après Android et iOS. <b>Le 9 octobre 2017, Microsoft annonce la fin du développement de l'OS, seules des mises à jour de sécurité seront développées.</b>
RIM Blackberry OS	BlackBerry	BlackBerry : téléphone intelligent et code propriétaire. BlackBerry 10 est la plate-forme de prochaine génération pour smartphones et tablettes BlackBerry, dont l'App Store a été lancé en Avril 2009. Tous les téléphones et les tablettes sont fabriqués par BlackBerry. Depuis, <b>elle a été l'une des plates-formes dominantes dans le monde, sa part globale du marché a été réduite à moins de 1% en fin 2014. L'Os de la plateforme est maintenant disponible en maintenance seulement.</b>

# Tableau de comparaison résumant les différences entre les trois différentes plateformes

	Virtual machine	Programming language	User interface	Memory management	IDE	Platform	Devices	App market
Android	Dalvik VM	Java	XML files	Garbage collector	Eclipse	Multi-platform	Heterogeneous	Google Play Store
iOS	No	Objective-C	Cocoa Touch	Reference counting	XCode	Mac OS X	Homogenous	iTunes Apps Store
Windows Phone 7	CLR	C# and .Net	XAML files	Garbage collector	Visual Studio	Windows Vista/7	Homogenous	Windows Phone Store

<http://www.sciencedirect.com/science/article/pii/S2090447915001276>

# Android ???? Histoire

- OS mobile
- Anecdote
  - **La société Android est allée voir en premier Samsung, qui a renvoyé gentiment les représentants de la société Android.**
  - Google à par la suite racheté Android en 2005, et à lancer son propre OS mobile.
- Selon le site android.com, le système est défini comme : « *La plate-forme la plus personnalisable et la plus facile à utiliser qui est installé dans plus d'un milliard dispositifs à travers le monde - des téléphones et les tablettes à des montres, de la télévision, de voitures et d'autres à venir*»

# Android <https://fr.wikipedia.org/wiki/Android>

## Rentabilité [ modifier | modifier le code ]

Android est rentable pour Google depuis le 5 octobre 2010<sup>90</sup> et son vice-président senior considère qu'Android aura rapporté plus d'un milliard de dollars de revenus à la fin de l'année 2010<sup>91</sup>.

D'après *Millennial Media*, Android génère plus de revenus publicitaires qu'iOS depuis octobre 2010<sup>92</sup>. Pour David Lawee, « l'achat d'Android est la meilleure affaire de Google »<sup>93</sup>.

Pour les développeurs d'applications, la rentabilité qu'offre Android est moins nette, surtout si on la compare avec l'offre de son concurrent l'*App Store* sur iOS. Si la plateforme Android a aujourd'hui plus de succès, si le nombre d'appareils est plus important, il est cependant plus difficile de transformer ce potentiel en ventes. Il semble que ce soit parce que les utilisateurs d'Android sont plus récents alors que les utilisateurs d'iOS sont une base établie, que le marché Android est plus libre alors que chaque compte iOS est directement associé à un moyen de paiement, alors que le système Android a une plus grande diversité qu'iOS<sup>94,95</sup>.

Dates	Activations quotidiennes
Décembre 2011	700 000 <sup>96</sup>
Juin 2012	1 000 000 <sup>97</sup>
Septembre 2012	1 300 000 <sup>98</sup>
Avril 2013	1 500 000 <sup>99</sup>
Mai 2015	3 500 000
Janvier 2016	?



Mascotte d'Android faite avec des canettes à New York.

# Android

- Lancé en fin 2008, a rapidement été adopté par de nombreux fabricants et est actuellement la plateforme avec la plus forte croissance dans le monde.
- Le succès d'Android est principalement dû à sa licence open source mais aussi le fait d'avoir un environnement de développement basé sur langage Java.

# Android

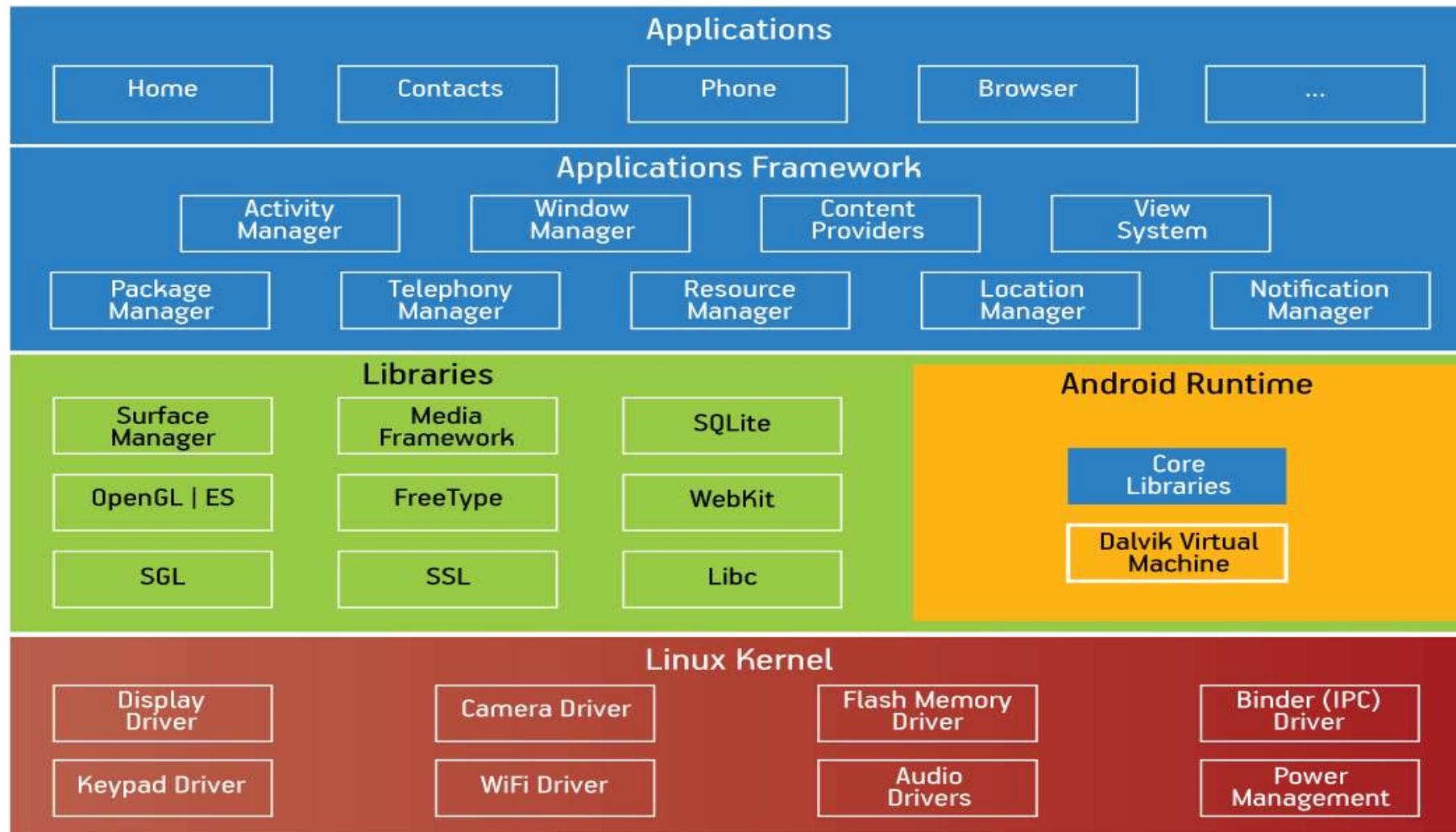
- Lancé initialement par Google, l'évolution d'Android est sous la responsabilité **l'Open Handset Alliance**, un consortium qui compte des dizaines de grandes entreprises du secteur tels que Google, Samsung, Intel, HTC, Motorola, etc. Ainsi l'OHA se compose :
  - d'opérateur mobile (Vodafone, Teleponica, Telecom Italia, China Mobile, etc.)
  - de fabricants de téléphone mobiles ( Asus, HTC, LG, Motorola, etc.)
  - de fabricants de semi-conducteur ( Intel, Nvidia, ARM, etc.)
  - d'éditeurs logiciels ( Ebay, Google, PacketVideo, etc.)
  - de distributeurs (Aplix corporation, Borqs, TAT)

# L'architecture du framework Android

- L'architecture d'Android est divisée en couches:
  - Le noyau Linux (couche inférieure) est la couche qui fournit les principaux services offerts par la plate-forme Android,
  - Les bibliothèques de classe
  - Le runtime d'exécution Android,
  - Le framework Application
  - Les applications, celle que nous apprendrons à développer tout au long de ce cours

Android est un stack logiciel pour terminaux mobiles qui inclut un système d'exploitation, du middleware et des applications clés.

# L'architecture du framework Android



# Platform usage

<https://fr.wikipedia.org/wiki/Android>

Popularité des différentes versions sur le Google Play Store\* le 2 octobre 2017<sup>36</sup>

\* Basée sur le nombre d'appareils Android qui se sont connectés au Google Play Store dans une période de sept jours se terminant à la date de collecte des données. Les versions avec un pourcentage inférieur à 0,1 % ne sont pas représentées ou comptées.

Version	Nom de code	Date de sortie	Version API	%
2.3.3 - 2.3.7	Gingerbread	6 décembre 2010	10	0,6 %
4.0.3 - 4.0.4	Ice Cream Sandwich	19 octobre 2011	15	0,6 %
4.1.x	Jelly Bean	9 juillet 2012	16	2,3 %
4.2.x		13 novembre 2012	17	3,3 %
4.3		24 juillet 2013	18	1,0 %
4.4	KitKat	31 octobre 2013	19	14,5 %
5.0	Lollipop	3 novembre 2014	21	6,7 %
5.1		9 mars 2015	22	21,0 %
6.0	Marshmallow	8 octobre 2015	23	32,0 %
7.0	Nougat	22 août 2016	24	15,8 %
7.1		4 octobre 2016	25	2,0 %
8.0	Oreo	21 août 2017	26	0,2 %



# Android: Pourquoi ce choix ?

- Android AOSP
  - Android Open Source Project: Possibilité de modifier Android comme l'on veut. (Nombreuses ROM disponibles)
  - Une large documentation
  - Un store d'applications le plus gros existant avec le Play Store
  - Coût financier 25 € à vie (Pour la publication sur le PlayStore) (100 € par an sur iOS)
  - Développement possible depuis n'importe quelle machine: MAC OS X, WINDOWS, LINUX, ANDROID

# iOS

- iOS, anciennement iPhone OS, est le système d'exploitation mobile développé par Apple pour plusieurs de ses appareils. Il est dérivé de macOS dont il partage les fondations.
- iOS comporte quatre couches d'abstraction, similaires à celles de macOS : une couche « Core OS », une couche « Core Services », une couche « Media » et une couche « Cocoa »<sup>2</sup>. Le système d'exploitation occupe au maximum 3 Go de la capacité mémoire totale de l'appareil, selon l'appareil.

iOS	
	
Famille	Unix
Langues	Multilingue (40 langues)
Type de noyau	Hybride
État du projet	Actif
Plates-formes	iPhone, iPad et iPod touch
Entreprise / Développeur	Apple
Licence	Propriétaire ; certaines parties sont disponibles sous <a href="#">Apple Public Source License</a>
États des sources	Logiciel propriétaire ( <a href="#">Darwin</a> est open source <sup>1</sup> )
Écrit en	C, C++, Objective-C, Swift et Java ↗
Dernière version stable	<a href="#">12.1.4 (16D57)</a> <sup>2</sup> (7 février 2019)
Dernière version avancée	iOS 12.2 bêta 2 (16E5191d) <sup>3</sup> (4 février 2019)
Méthode de mise à jour	<a href="#">iTunes</a> / OTA
Environnement de bureau	Cocoa Touch
Gestionnaire de paquets	<a href="#">App Store</a> ↗
Site web	<a href="http://www.apple.com/fr/ios/ios-12/">www.apple.com/fr/ios/ios-12/</a> ↗ [archive]

# iOS

- Objective-C depuis le début
- JavaScript , C# , C++ , Java , Swift, Cocoa
- App Store : 2.2 millions d'applications, 140 milliards de téléchargements depuis le début et deux tiers des applications gratuites
- L'environnement de développement n'existe que sur MacOS
  - Nécessite un Mac, ainsi qu'un device iOS
  - La dernière version de MacOS est généralement indispensable



# Applications mobiles

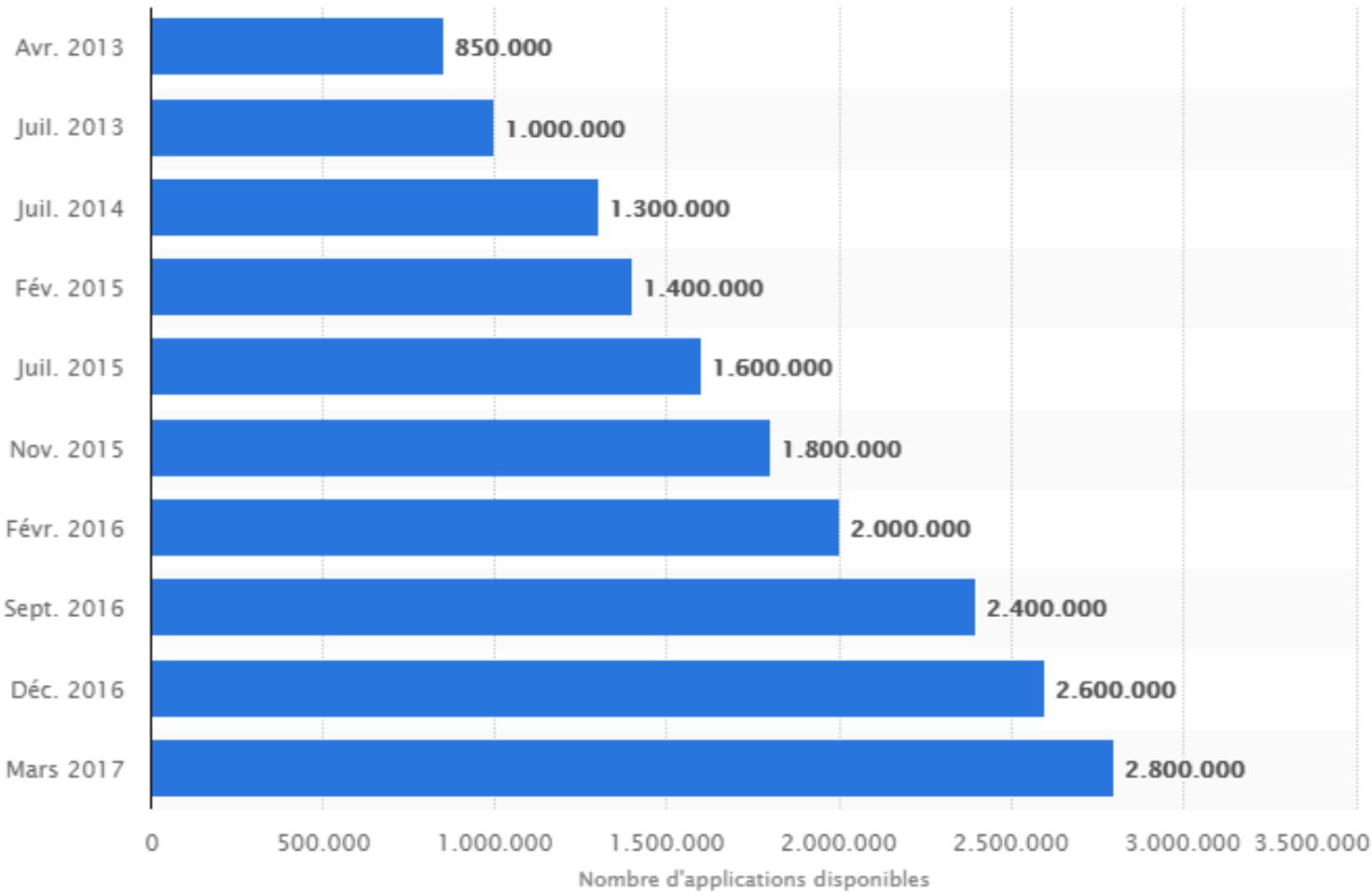
- Une application mobile est un logiciel applicatif développé pour un appareil électronique mobile, tel qu'un assistant personnel, un téléphone portable, un « smartphone », un baladeur numérique, une tablette tactile, ou encore certains ordinateurs fonctionnant avec le système d'exploitation Windows Phone.
- Elles sont pour la plupart distribuées depuis des plateformes de téléchargement (parfois elles-mêmes contrôlées par les fabricants de smartphones) telles que l'App Store (plateforme d'Apple), Play Store (plateforme de Google / Android), ou encore le Windows Phone Store (plateforme de Microsoft).

# Écosystème mobile en 2016

- En 2009, le nombre de téléchargements d'applications mobiles sont élevées à environ 2,52 milliards de dollars US
- En 2010, les applications mobiles ont rapportés 6,8 milliards de dollars.
- En 2018, plus de 194 milliards de nouvelles applications ont été téléchargées sur l'App Store et Google Play

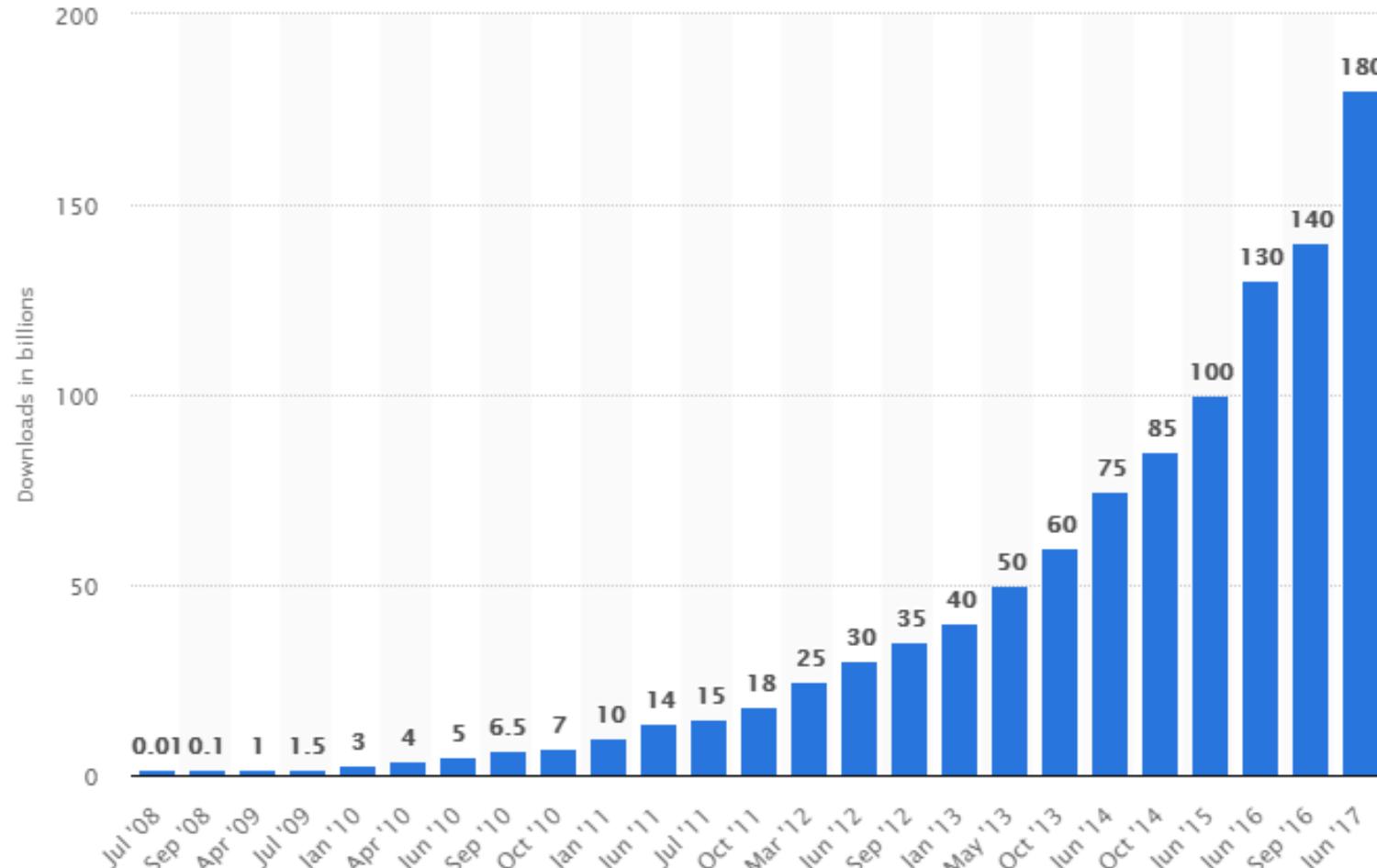


# Nombre d'applications disponibles sur Google Play Store de avril 2013 à mars 2017



<https://fr.statista.com/statistiques/565393/nombre-d-applications-disponibles-sur-google-play-store/>

# Nombre d'applications disponibles sur App Store de Juillet 2008 à Juin 2017 en billions



<https://fr.statista.com/statistiques/565393/nombre-d-applications-disponibles-sur-google-play-store/>

# Comparaison entre les plateformes

## iOS

- +Application vérifiée, et contenant moins de virus que les applications Android
- -Prix d'une License (100€/an)
- -Nécessaire d'avoir un MAC
- IDE : Xcode
- Langage : Objective C et Swift

## Android

- +Grande communauté d'utilisateurs et de développeur
- +Prix d'une License (25€/vie)
- - Compatibilité non assurés pour les anciennes versions
- IDE : Android Studio / Eclipse / NetBeans
- Langage : HTML5/CSS3/JavaScript ou Java, Kotlin

## Windows Phone

- - Peu d'utilisateur
- - Compatible uniquement avec marque Lumia
- Ide : Visual Studio
- Langage : C#

# Travaux dirigés

- Définissez un appareil mobile
- Rappelez les principales fonctionnalités des appareils mobiles des décennies 1980-1990, 1990-2000 ?
- Quelles sont les évolutions qu'il y a eu durant les années 2000 sur les appareils mobiles ?
- Qu'est-ce qu'un smartphone ?
- Citer quelques marques de Smartphone
- Qui domine le marché des smartphones ?
- Qui a mis le premier smartphone dans le marché ? Quand ?
- Citer quelques fonctionnalités d'un smartphone

# Travaux dirigés

- Définissez un système d'exploitation mobile
- Rappelez les principales fonctionnalités d'un système d'exploitation mobile ?
- Décrire l'évolution de la part de marché de chaque OS Mobile ?
- Dresser la situation en début 2017 de chaque OS Mobile
- Quelle est la relation entre les plates-formes mobiles, l'environnement de développement et un market ?
- Quels sont les critères de choix d'une plateforme ?
- Quels sont les principales plateformes mobiles en 2017 ?
- Qu'est-ce qui fait le succès d'Android ?

# Travaux dirigés

- Qu'est ce qui a été à l'origine des échecs des plateformes Windows Phone et BlackBerry en vous référant à cet article  
<Http://www.zdnet.fr/actualites/blackberry-os-et-windows-phone-abandonnes-par-de-grands-noms-des-applis-39834388.htm> ?
- Comment sont distribuées les applications mobiles ?
- Quelle est l'entreprise qui a créée Android ?
- Sous quelle licence est distribuée Android ?
- Quelle est la structure qui gère l'évolution d'Android ?
- Quel est le nom de la dernière version de l'OS Android ?
- Décrire l'architecture du Framework Android

# Travaux dirigés

- Selon vous pourquoi la version Marshmallow est-elle la version la plus utilisée actuellement ?
- Dresser une comparaison entre les plateformes
- Qu'est qui motiverait le choix en Android et iOS ?
- Lire l'article du journal LeMonde et proposer un résumé  
[http://www.lemonde.fr/economie/article/2017/01/17/le-commerce-des-applications-mobiles-toujours-plus-rentable\\_5063746\\_3234.html](http://www.lemonde.fr/economie/article/2017/01/17/le-commerce-des-applications-mobiles-toujours-plus-rentable_5063746_3234.html)



TypeScript



# INF 3511 Programmation des Mobiles: Environnements Mobiles



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

Partie 3 :  
Approches de développement mobile



# Approches de développement mobile

# A propos de moi

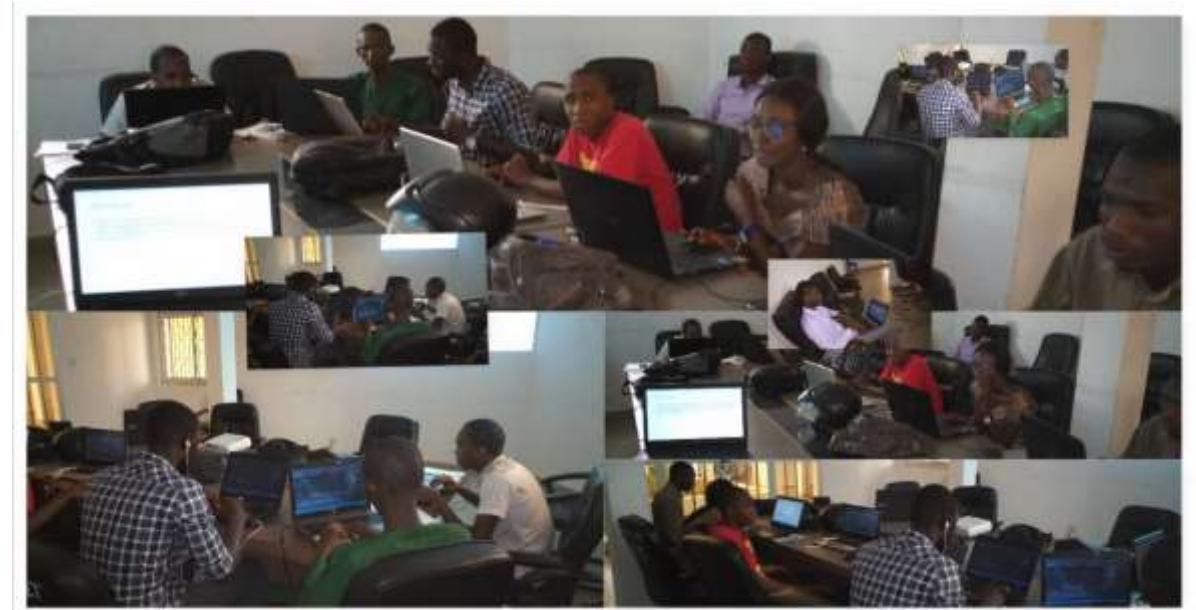


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
- 3. Approches de développement mobile**
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android

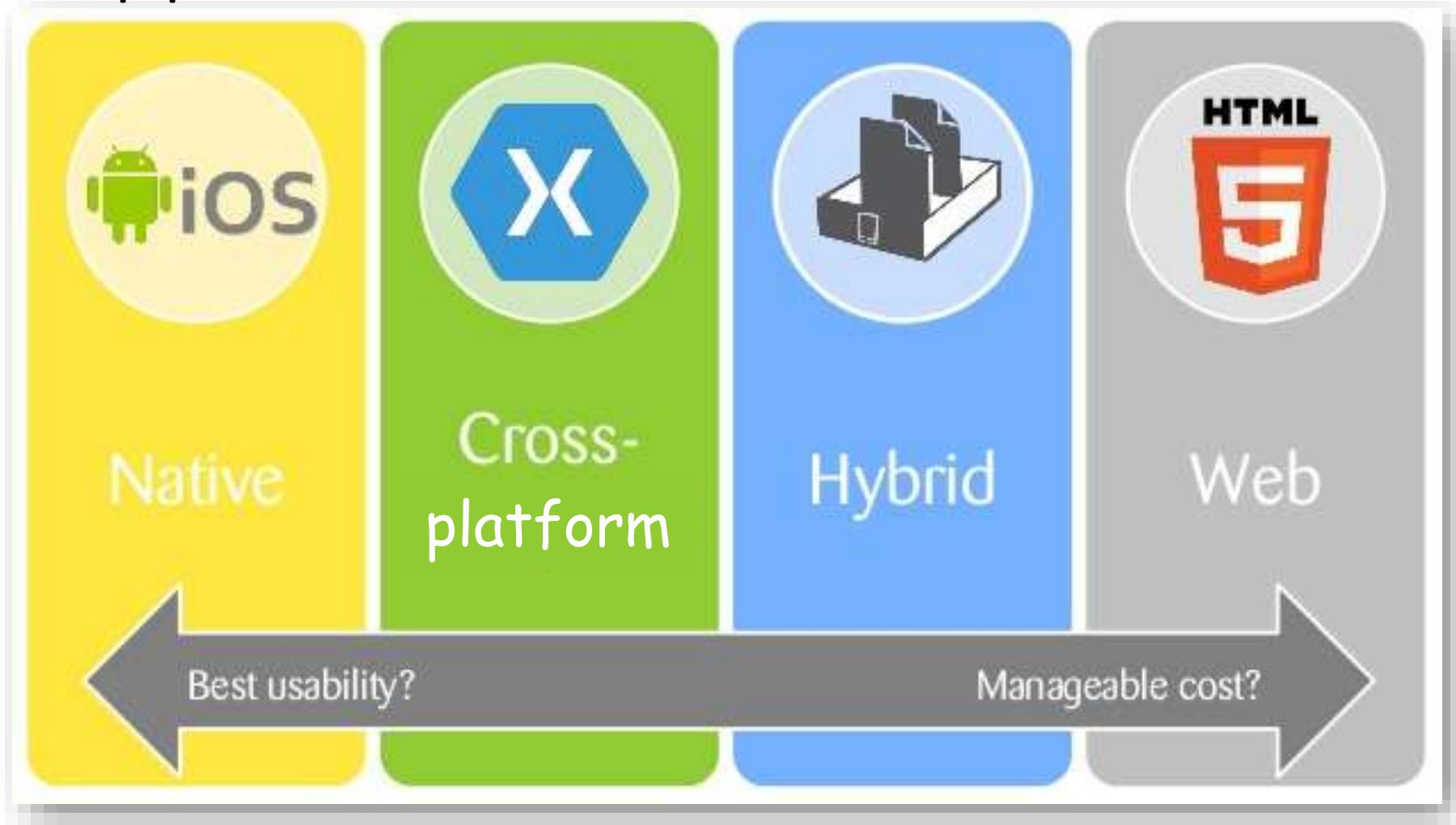


# Différentes approches pour le Développement Mobile



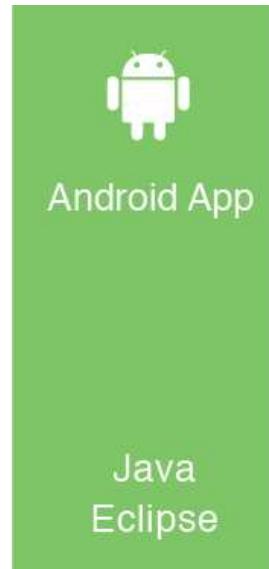
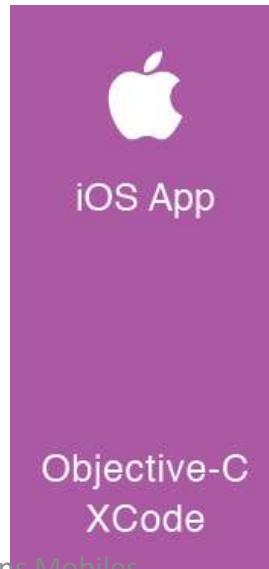
- **Plusieurs options pour développer une application mobile :**
  - **Développement natif** (Objective C ou swift, java, C#...)
  - **Solutions hybrides** : Des technologies de type Phonegap ou Cordova permettent d'embarquer une web view dans une application native. Ceux-ci permettent d'utiliser les fonctionnalités natives des smartphones. De plus elle pourra être distribuée en tant qu'application sur les plateformes d'applications (App Store, Android Market, etc.).
  - **Développement web**: utilisant le HTML5, CSS, JavaScript,...
  - **Solutions cross plateformes(natives)** : solutions comme Xamarin permettent de construire des applications iOS, Android et Windows mobile dans un environnement de développement Microsoft (C#)

# Différentes approches pour le Développement Mobile



# L'approche native: L'Approche en Silo

- **Les applications natives** sont spécifiques à une plate-forme mobile donnée (iOS ou Android) utilisant les outils de développement et le langage pris en charge par la plate-forme (par exemple Xcode et Objective-C avec iOS, Eclipse et Java avec Android). Les applications natives ont un design plus apprécié et fonctionnent plus performant.
- **Créer l'application plusieurs fois**
  - Plus de développeurs
  - Plus de maintenance
  - Différents environnements



# L'approche native: L'Approche en Silo

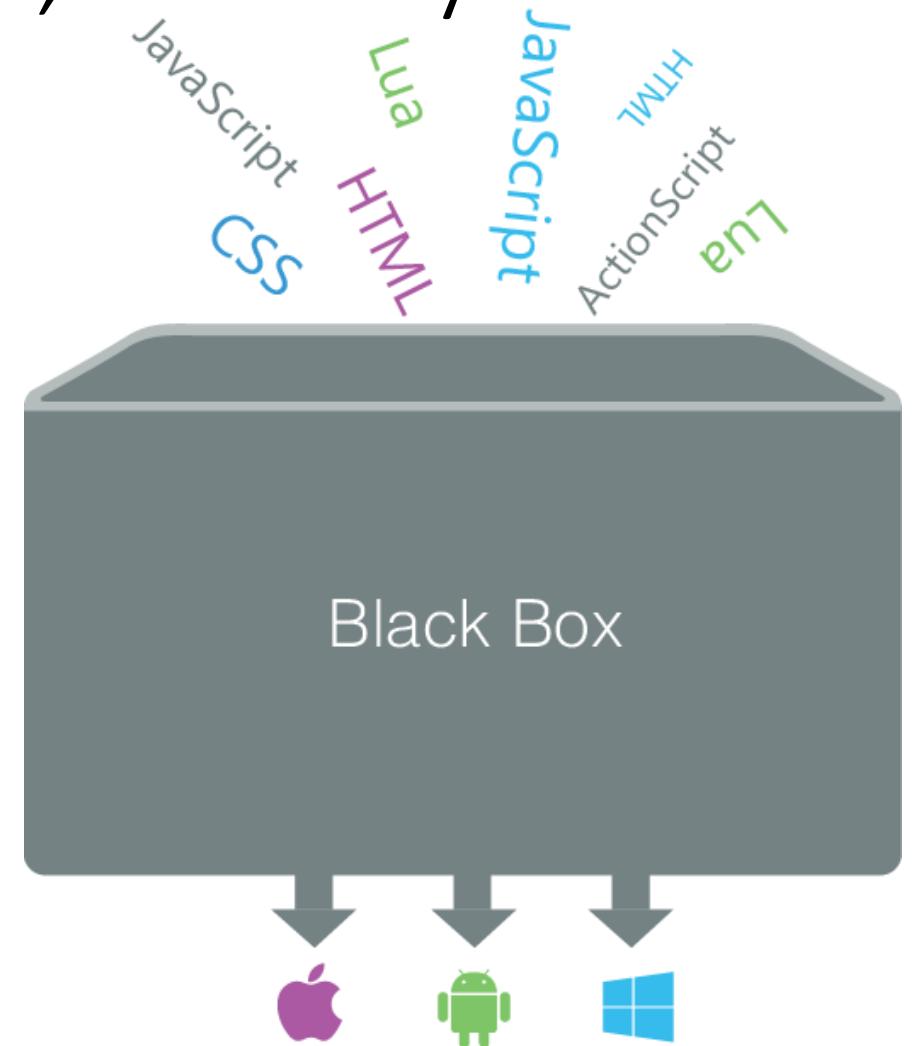
- **Avantages :**
  - Chaque système d'exploitation a sa version native – Possibilité de faire appel aux fonctionnalités de l'appareil utilisé (telle que la camera, la géolocalisation, le gyroscope ...) et ainsi les exploiter de manière infinie...
  - Profiter d'une visibilité sur les stores d'applications et optimisation du référencement
  - Accès aux APIs natives – Plus simple à développer et plus d'outils à disposition et de support en cas de problème
  - Intégration à la facturation des stores si l'objectif est de rentabiliser le produit (Apps payantes)
  - Facilité à se conformer aux ergonomies spécifiques de chaque système d'exploitation afin d'avoir la meilleure expérience utilisateur possible.

# L'approche native: L'Approche en Silo

- **Inconvénients :**
  - Le cout de développement est important car il faut développer une version spécifique pour chaque système d'exploitation
  - Les performances sont très souvent meilleures en natif
  - Toutes les mises à jour nécessiteront un coût de développement supplémentaire multiplié par le nombre de plate-formes utilisées

# L'approche web: Write Once, Run Anywhere

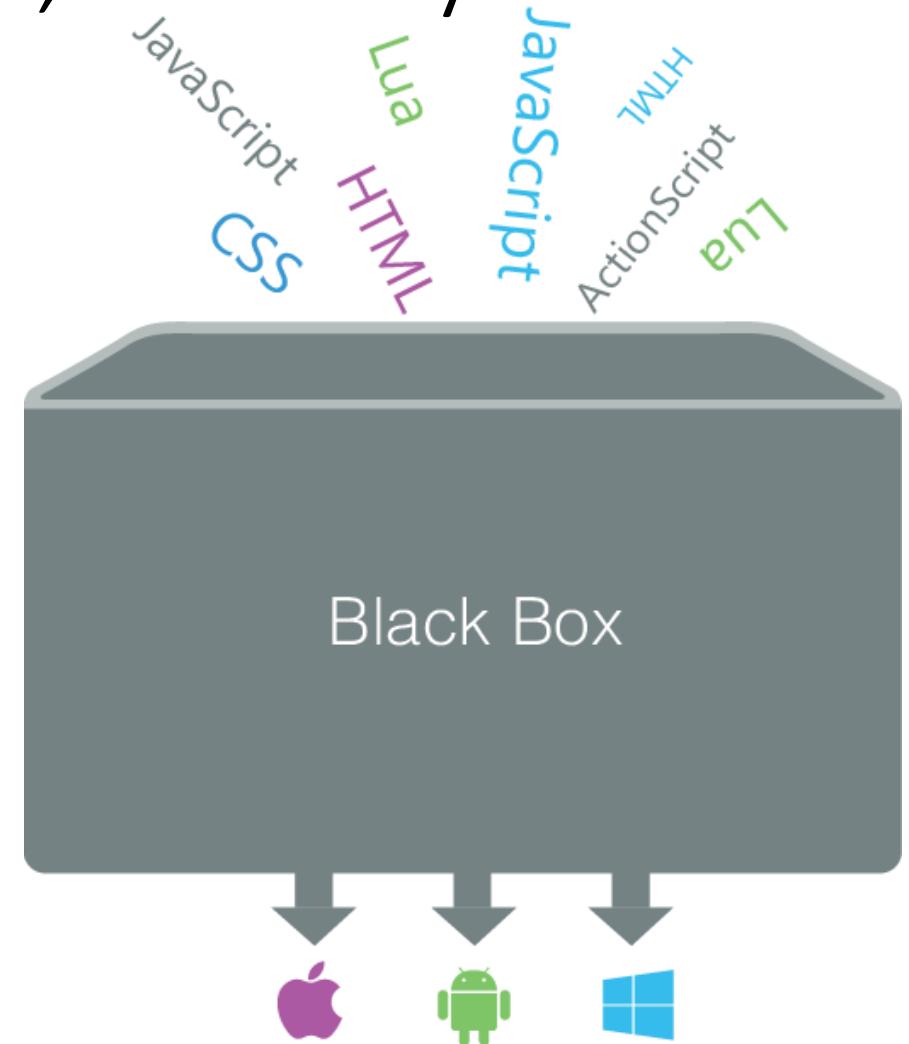
- Utilise des technologies Web standard, généralement HTML5, JavaScript et CSS. Développée à grand renfort de codes javascript et autres boites à outils de type Angular ou [React](#)
- Approche de développement mobile permettant de créer des applications mobiles multiplateformes fonctionnant sur plusieurs appareils.



# L'approche web: Write Once, Run Anywhere

- **Avantages :**

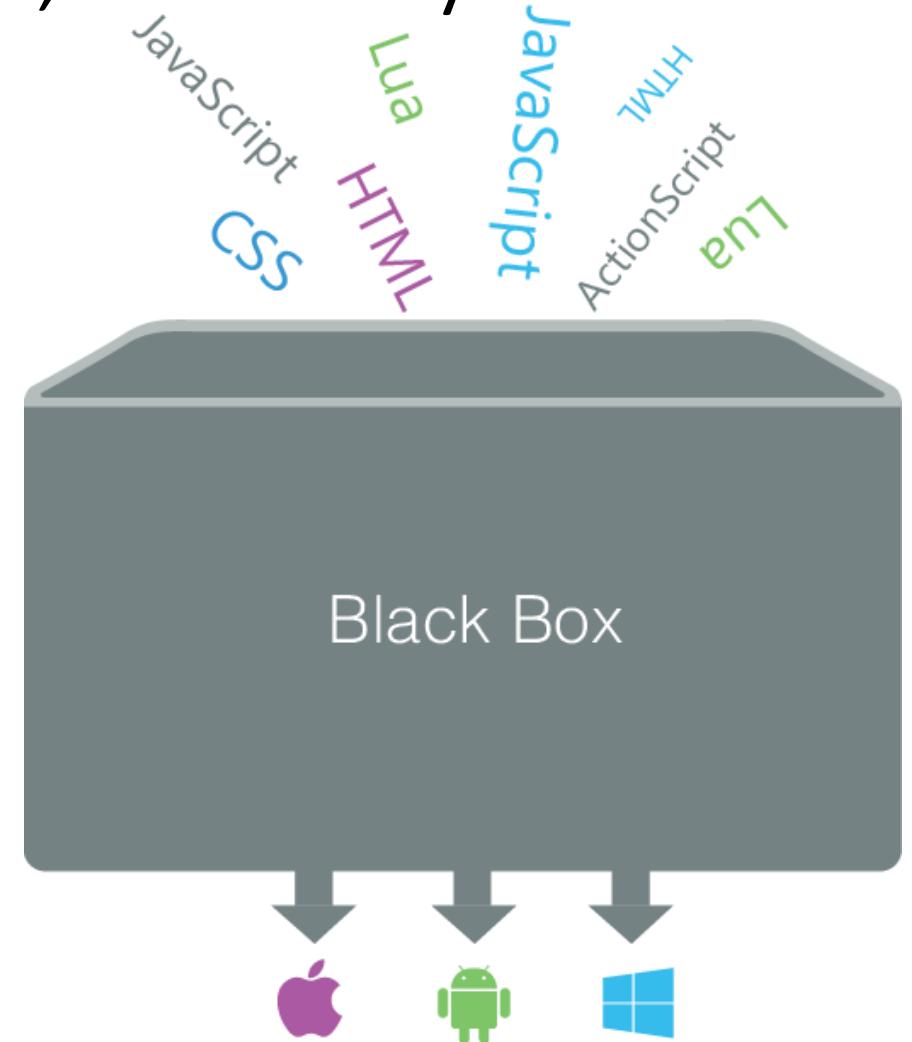
- Développement web classique donc très bien maîtrisé par les développeurs.
- Utilisation de HTML5, CSS3, JavaScript et de frameworks – Responsive design pour un affichage optimal et adapté aux différents supports (smartphones, tablettes, desktop...) – Facilité de mise à jour du contenu



# L'approche web: Write Once, Run Anywhere

- **Inconvénients :**

- Nécessité d'avoir une connexion internet pour accéder au contenu – Pas de visibilité sur les stores d'application
- Accessible via le navigateur. -Pas d'icone d'application – Ne s'exécute pas en plein écran



# Application native vs. Application Web

- Développement spécifique selon le navigateur web
- Manque d'intégration avec l'OS(capteurs, partage, etc)
- Expérience utilisateur
- Performance « dégradées »

Facebook Web



Facebook natif



# Approche hybride

Application utilisant le [navigateur web](#) intégré du support ([Smartphone](#) ou [tablette](#)) et les technologies Web ([HTML](#), CSS et Javascript) pour fonctionner sur différents OS ([iOS](#), [Android](#), [Windows Phone](#), etc.). Une telle application utilise les fonctionnalités natives des Smartphones et peut être distribuée sur les plateformes d'applications telles que [l'AppStore](#), le [Google Play](#), etc. [https://fr.wikipedia.org/wiki/Application\\_hybride](https://fr.wikipedia.org/wiki/Application_hybride)

- Une application hybride est comme une application Web, principalement construite à l'aide de HTML5 et de JavaScript, puis encapsulée dans un conteneur natif mince qui donne accès aux fonctionnalités de la plate-forme native.
- Il existe de nombreux frameworks mobiles hybrides tels que Ionic, NativeScript, React Native, Xamarin, PhoneGap etc. PhoneGap est un exemple de conteneur le plus populaire pour créer des applications mobiles hybrides.
- Le développement hybride combine le meilleur (ou le pire) des mondes natif et web.



# Approche hybride

- Cependant la qualité, la performance, et la résolution de ces applications sont nettement inférieures à celles des applications natives.
- En effet l'application hybride peut ne pas bien s'adapter au système d'exploitation utilisé par le smartphone du mobinaute (interface polluée par des widgets inutiles, mauvaise résolution etc.).
- En plus les applications hybrides ne sont accessibles que sur iPhone et Android, et sont parfois refusées sur certaines plateformes d'applications.

# Approches multiplateformes

The developers use the cross-platform mobile development solutions to develop the mobile application once and run it on many platforms.

- Il n'existe pas d'interopérabilité entre les langages utilisés pour le développement mobile ciblant les différentes plateformes.
- Créer des applications Android, iOS et Windows Phone - de la logique métier à l'interface utilisateur - avec un code presque 100% commun comme C# sous Xamarin qui sera généré une application en code natif de la plateforme cible.

Le marché des applications multiplateformes devrait atteindre 7,5 millions de dollars d'ici 2018, et le nombre d'outils de développement multiplateforme est en hausse. Source <https://blog.octo.com/etat-de-lart-des-solutions-cross-platform-mobile/>

Pour les catégories d'app cross-plateformes. Voir :  
<http://www.sciencedirect.com/science/article/pii/S2090447915001276>



# Quatre principales approches multiplateformes: Web, Hybrid, Interpreter and Cross-Compiler

- **Approche Web:** Une application créée avec l'approche Web est essentiellement un site Web accessible via le navigateur Web du téléphone. L'application est créée avec HTML, CSS et Javascript et est téléchargée pour fonctionner sur le smartphone.
- **Approche Hybride:** Hybride utilise le même concept que l'approche Web, mais n'exécute pas l'application dans un navigateur Web mais dans un conteneur natif sur le périphérique (fourni par le framework).

# Quatre principales approches multiplateformes: Web, Hybrid, Interpreter and Cross-Compiler

- **Approche Interprétée(Interpreter):** une application interprétée est téléchargée sur le périphérique des utilisateurs et l'interpréteur décide à l'exécution quel code doit être exécuté en fonction du périphérique en cours. L'approche interprétée utilise une couche d'abstraction pour permettre l'accès aux entités natives.
- **Approche Cross-Compiler:** Cross-Compiler prend du code et sort du code natif ou des binaires pour chaque plate-forme spécifique. Le code natif donné est ensuite compilé à nouveau sur les différentes plates-formes. L'application dépend donc de l'efficacité du Cross-Compiler.

# Approche multiplateformes

- **Avantages :**
  - Chaque système d'exploitation a sa version native – Profiter d'une visibilité sur les stores d'applications – Les modifications et mises à jour seront effectives sur chaque plate-forme – Economie de budget – Idéale pour des applications de jeu en 2D ou en 3D
- **Inconvénients :**
  - Problème d'accès aux APIs – Problème de validation des applications par les stores – Difficile à maintenir et à faire évoluer – Toutes les fonctionnalités des appareils ne peuvent pas encore être exploitées – Problème d'ergonomie. UX et UI non optimisées

# Xamarin

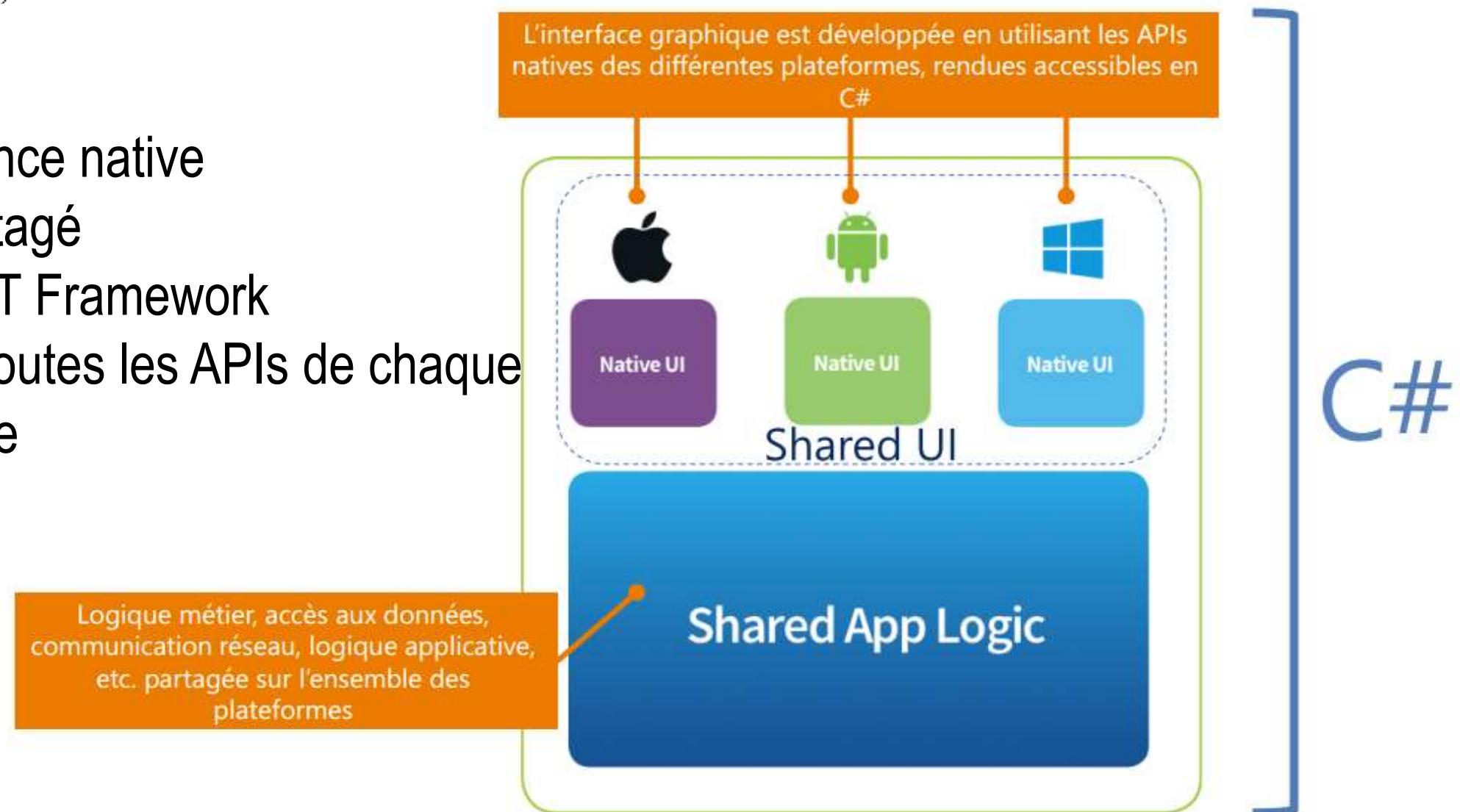
Xamarin, qui appartient à Microsoft, utilise son propre IDE pour le développement (Visual Studio sur Windows et Xamarin Studio sur OSx). La langue utilisée est C # et est conçue en utilisant leur IDE. Le code est compilé en code natif pour iOS et un fichier APK natif (Android Application Package). En 2017, Xamarin est au moins utilisé par 1,4 million de développeurs autour du monde et a d'autres applications dans sa suite pour les tests, etc.

- Xamarin est une société américaine fondée en mai 2011
- S'appuie sur le développement de la technologie mono
- Mono est créé en tant que projet open source, en vue de proposer une implémentation de la plate-forme .Net sous UNIX
- En février 2016, Xamarin est rachetée par Microsoft
- Mise à disposition gratuite de Xamarin dans Visual Studio Community Edition.



# Approche multiplateformes: L'approche de Xamarin

- UI native
- Performance native
- Code partagé
- C# & .NET Framework
- Accès à toutes les APIs de chaque plateforme



# Défis du développement d'applications mobiles

- Un paysage de fournisseurs concurrentiel et fluide (Apple, le consortium Android, notamment Amazon, RIM, HP) signifie que les applications doivent être multiplateformes pour une large adoption
- Pas d'appareil "standard" (qu'en est-il des appareils iOS, Windows Phone?)
- Faible entrée de bande passante (dans la plupart des cas, qu'en est-il des tablettes?)
- Taille d'écran limitée (tablettes?)
- Un manque de fiabilité de la connectivité et du périphérique (accès réseau, alimentation, lumière ambiante, bruit, au moins pour l'instant)
- Intégration des compromis avec les services cloud et d'entreprise

# Supports du développement d'applications mobiles

- Langages orientés objet de troisième génération (iOS - Objective C ou Swift, Android - Java, Windows Phone - C #)
- Langages de script (JavaScript, Ruby)
- Structures multiplateformes - Titane, RhoMobile, Xamarin, PhoneGap
- Intégré dans des "frameworks" spécifiques pour le développement d'applications mobiles

# Points clés

- iOS et Android ont participé à réduire la **fragmentation de l'industrie mobile** et ont attiré ainsi rapidement de nombreux développeurs
- iOS et Android sont à l'origine de l'**essor des smartphones et des marchés de l'Internet mobile**, tandis que les tentatives précédentes de type WAP et de « Walled gardens » ont échoué.
- Ce contexte a été favorable au développement d'**une économie dynamique des applications mobiles en Europe**, avec près de 1,5 million d'emplois et 13 milliards EUR de revenus générés (applications payantes, publicités) en 2016.

<https://fr.idate.org/evolution-de-lecosysteme-mobile/>

# Points clés

- De nombreuses **success stories** ont été rendues possibles indépendamment de iOS et Android, notamment dans le domaine des jeux sur mobile (ex. Angry Bird de Rovio) ou dans l'écosystème de Facebook.
- **La concurrence entre les acteurs des terminaux a été renforcée**, plus intense qu'avant le lancement de iOS et Android. Android en particulier ayant notamment favorisé l'arrivée de nouveaux entrants grâce à un abaissement des prix.
- **Les développeurs utilisent couramment l'approche hybride ou multi-plateformes**, notamment en tirant profit des outils cross-platform.

<https://fr.idate.org/evolution-de-lecosysteme-mobile/>

# Travaux dirigés

- Qu'est-ce qu'une approche de développement ?
- Lister les approches de développement mobile
- Décrire les approches de développement mobile
- Décrire brièvement chaque approche
- Listes 03 avantages et 03 inconvénients de chaque approche
- Expliquer l'approche multiplateforme et donner quelques raisons liées à son succès
- Lister 5 défis du développement mobile
- Quels sont les supports du développement d'applications mobiles ?

# Travaux dirigés

- Quelles sont les motivations liées au développement Cross-Platform ?
- Qu'est-ce-que Xamarin?
- Qu'est qui a été à l'origine de l'essor des marchés de l'Internet mobile ?
- Citer une des applications mobiles de jeux qui connaît un success-story actuellement au Sénégal
- Faire un résumé des points clés sur les plateformes mobiles

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.



TypeScript



# INF 3511 Programmation des Mobiles: Environnements Mobiles



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

**Partie 4:**

**Xamarin pour le développement d'Applications Mobiles**



# A propos de moi

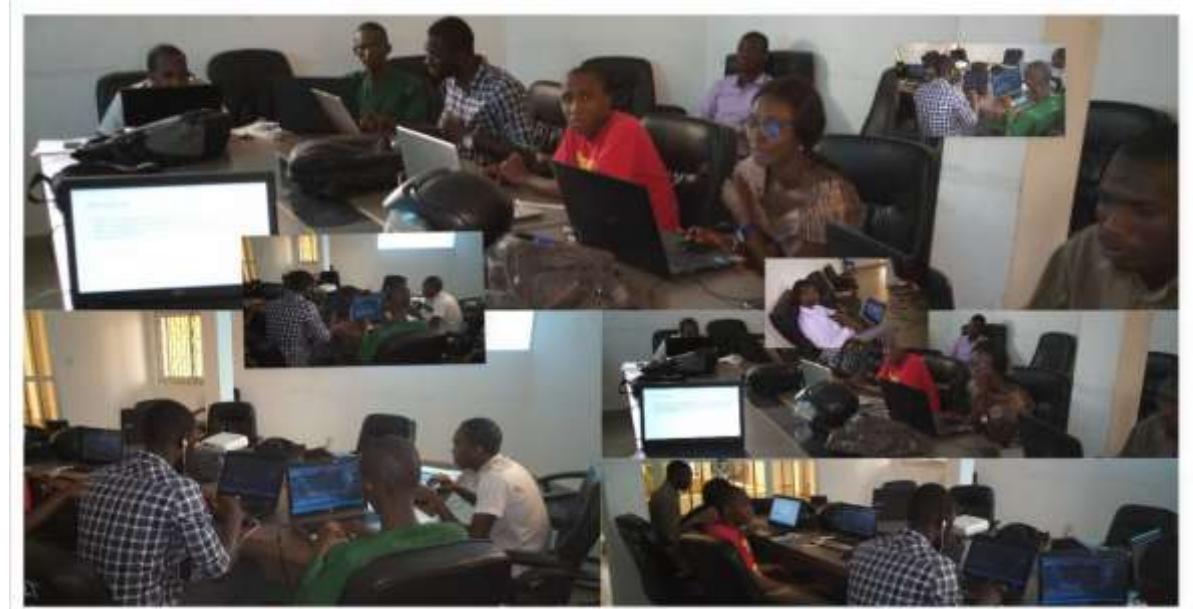


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

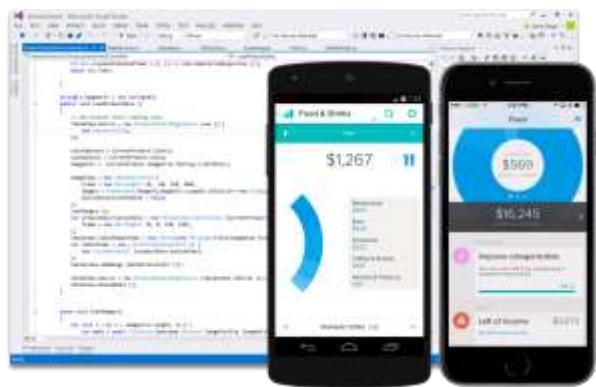
1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
3. Approches de développement mobile
- 4. Xamarin pour le développement d'Applications Mobiles**
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. Développement d'Applications Natives avec Android



# Xamarin

- Comme évoqué précédemment, Xamarin est un framework .Net de développement d'applications mobiles natives ou cross-plateforme, fondé en 2011 et racheté par Microsoft en 2016. L'avantage de Xamarin est de fournir une facilité de communication avec les produits microsoft (Azure, Sql server, etc).
- Cette technologie permet de développer des applications mobiles pour les plateformes suivantes :
  - Android
  - iOS
  - Windows Universal (qui comprend Windows Phone)
- Le développement Xamarin comprend deux approches : le développement natif, et le développement cross-platform.

# Xamarin est un moyen de



Créer



Tester



Monitorer

Xamarin

**Xamarin**

Creation: mai 2011

Fondateurs: Des anciens membres de Novell travaillant sur Mono<sup>1</sup>

Personnages clés: Miguel de Icaza, Nat Friedman

Forme juridique: Inc.

Siège social: Cambridge, Massachusetts (États-Unis)

Direction: Nat Friedman (CEO), Miguel de Icaza (CTO), Joseph Hill (COO)<sup>2</sup>

Actionnaires: Microsoft

Activité: Logiciel

Produits: Mono (stewardship et support), Mono for Android, MonoTouch

Société mère: Microsoft

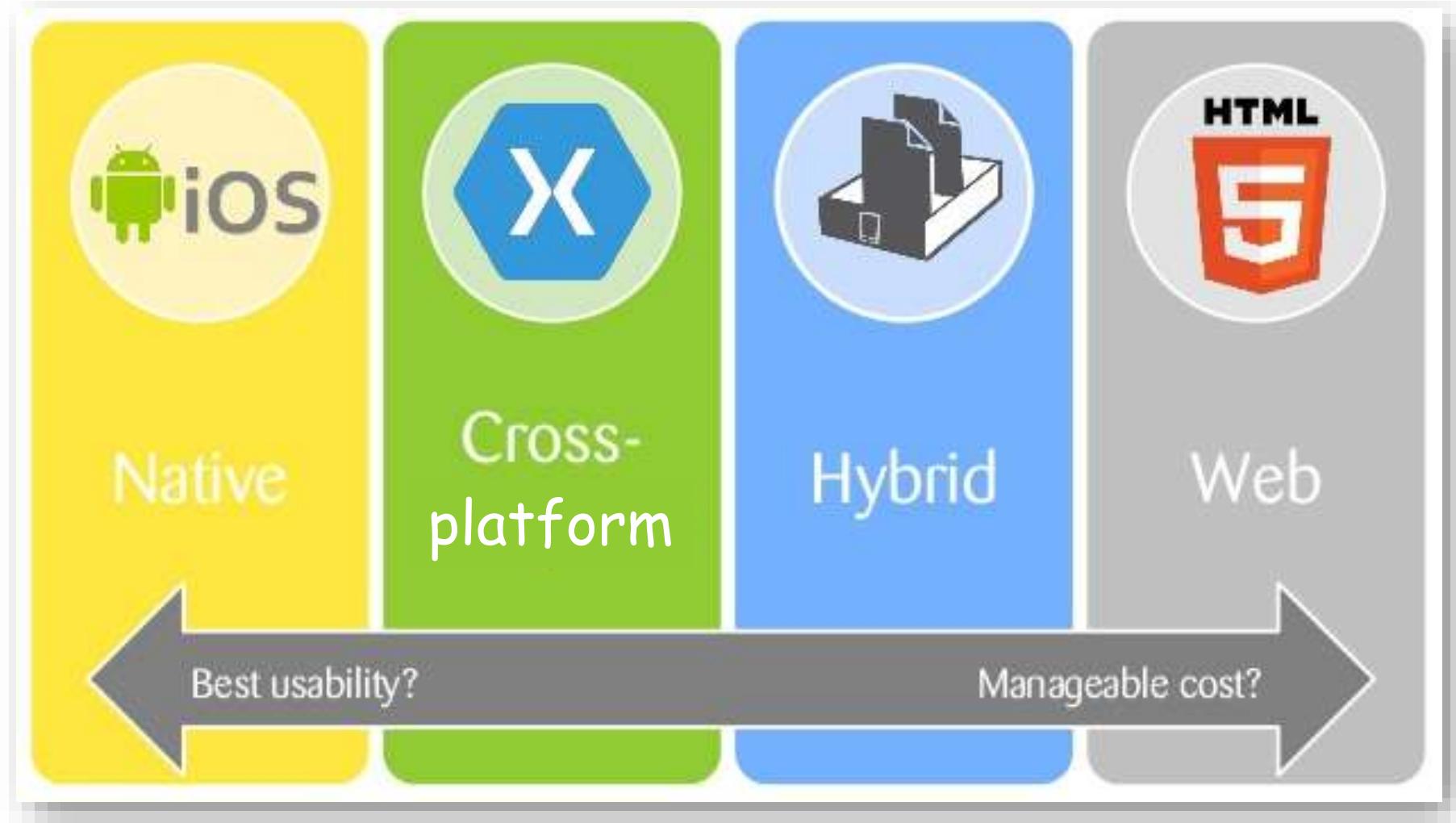
Effectif: 50<sup>3</sup>

Site web: [www.xamarin.com](http://www.xamarin.com) [archive]

modifier - modifier le code - voir wikidata

# Différentes approches pour le Développement Mobile: rappel

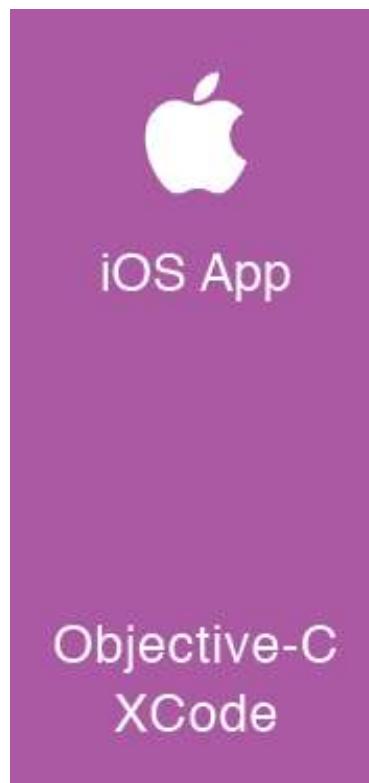
# Approches pour le Développement Mobile



# L'approche native: Silo Approach

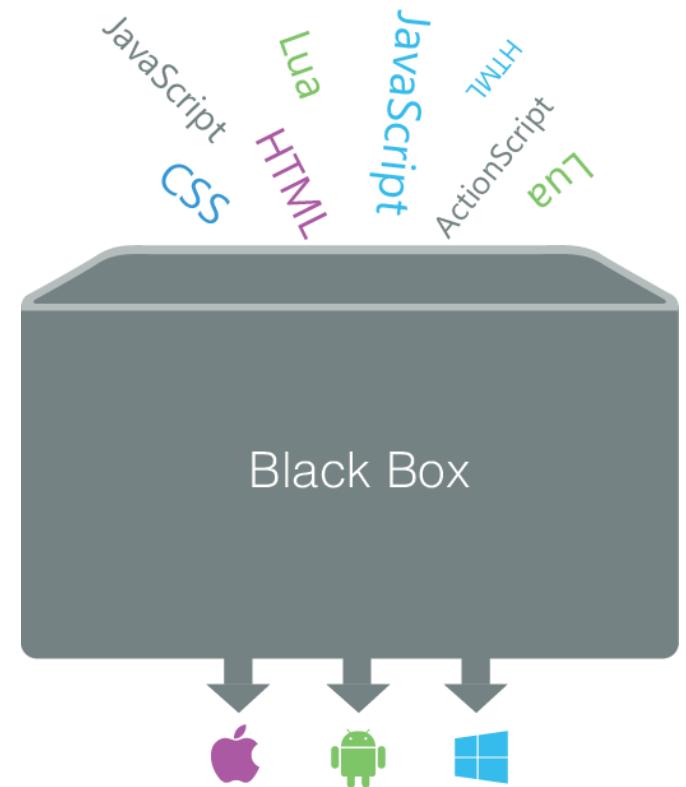
- Créer l'application plusieurs fois

- Plus de développeurs
- Plus de maintenance
- Différents environnements



# L'approche Write Once, Run Anywhere

- Nécessaire d'avoir une connexion Internet en tout temps. Si l'utilisateur n'est pas connecté à Internet avec une utilisation illimitée des données, l'application peut consommer une quantité considérable de données en fonction de l'application.
- La vitesse de connexion Internet peut également limiter l'expérience de l'application.
- L'application peut faire un certain usage du matériel du téléphone à travers diverses interfaces disponibles dans le navigateur web mais est souvent limitée par ce que le fournisseur du navigateur a implémenté



# L'approche hybride

- Application utilisant le navigateur web intégré du support (Smartphone ou tablette) et les technologies Web (HTML, CSS et Javascript) pour fonctionner sur différents OS (iOS, Android, Windows Phone, etc.).
- Une telle application utilise les fonctionnalités natives des Smartphones et peut être distribuée sur les plateformes d'applications telles que l'AppStore, le Google Play, etc.



# L'approche hybride: inconvénients

- le code source est mélangé et la compilation se fait en de multiples étapes, ce qui peut fragiliser le fonctionnement de l'application mobile et rendre la maintenance longue et complexe ;
- Tout n'est pas réalisable en terme de fonctionnalités avec cette technologie, il faut s'assurer en amont de la compatibilité de votre projet (et de ses évolutions futures !) avec les technologies hybrides. Par exemple l'application peut ne pas supporter toutes les fonctionnalités.
- Des contenus trop lourds peuvent avoir de plus forts impacts sur les performances qu'avec une application native ;
- Le mode hors-ligne est plus limité et plus délicat à concevoir ;
- Enfin, en utilisant une application hybride, nous devenons dépendant d'un logiciel tiers – en l'occurrence Cordova – et donc de sa compatibilité avec les nouvelles versions d'OS.

<http://www.mobizel.com/2015/08/developpement-dune-application-mobile-hybride-33/>

# Application native vs. Application Web

- Développement spécifique selon le navigateur web
- Manque d'intégration avec l'OS(capteurs, partage, etc)
- Expérience utilisateur
- Performance « dégradées »
- Beaucoup ont essayé le web pour finalement revenir au natif

Facebook Web



Facebook natif



# Approche cross-platform

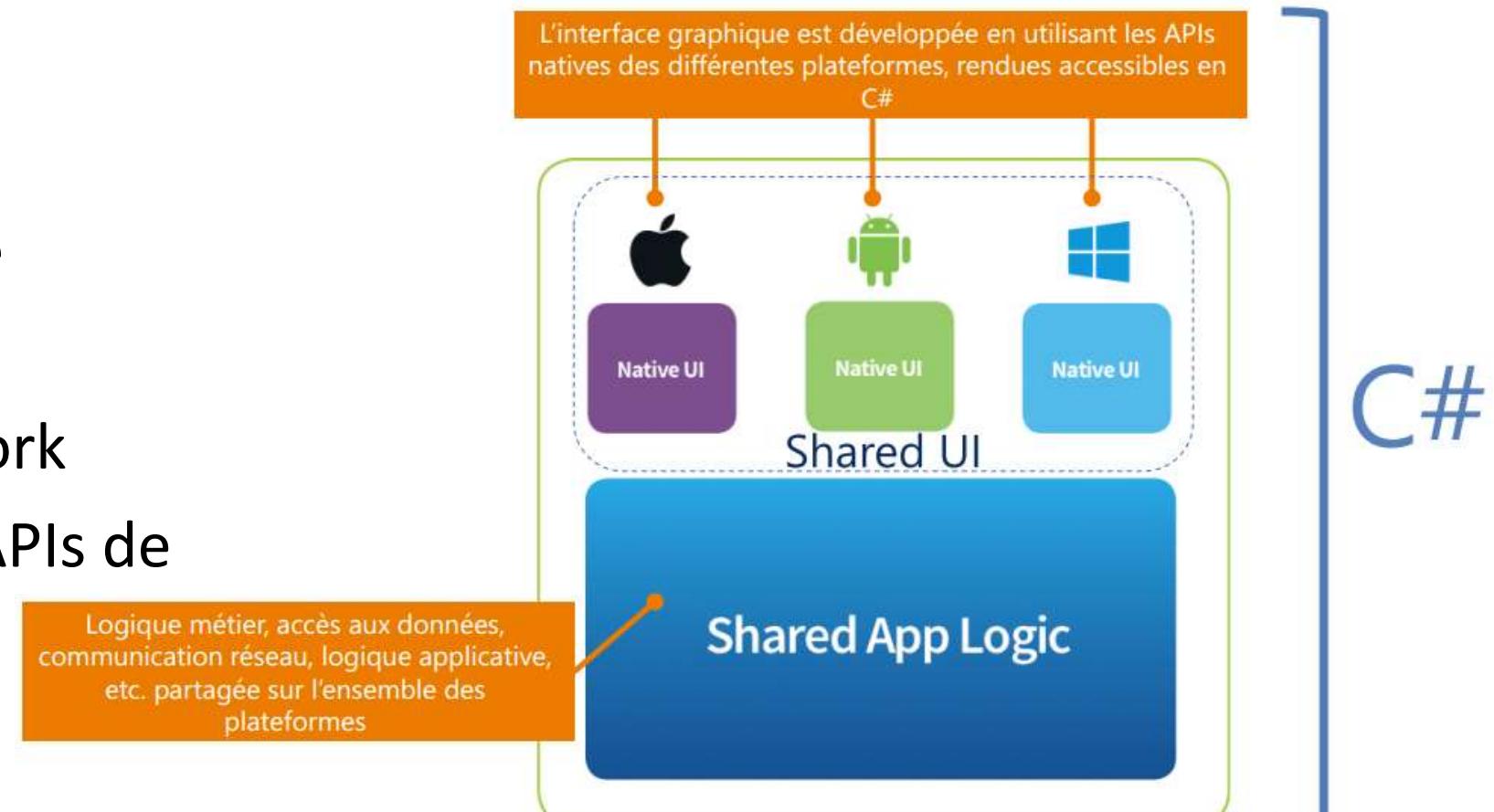
Quatre principales approches multiplateformes: Web, Hybrid, Interpreter and Cross-Compiler



# L'approche de Xamarin

- UI native
- Performance native
- Code partagé
- C# & .NET Framework
- Accès à toutes les APIs de chaque plateforme

Bâtir une app native iOS, Android et Windows à partir d'une base de code C# unique.



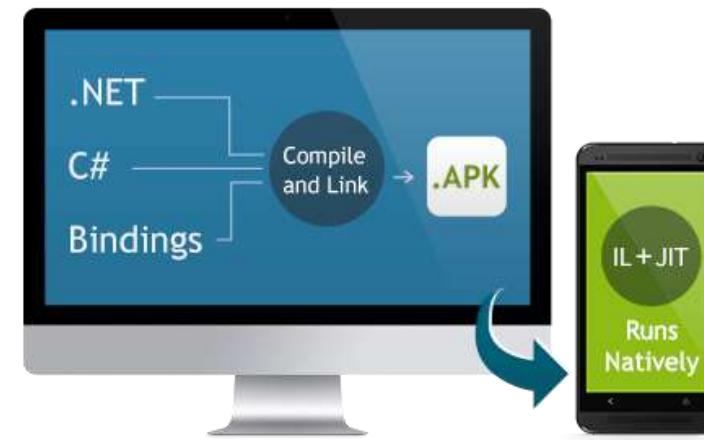
Xamarin recently introduced Xamarin.Forms a new library for cross platform user interface. We will touch up on this later, but this enables you to be highly productive, share code, but build out UI on each platform and access platform APIs. With Xamarin.Forms you now have a nice Shared UI Code layer, but still access to platform APIs. You can start from native, pick a few screens, or start with forms, and replace with native later

# Performance native

Tout ce que vous pouvez faire en Objective-C, Swift ou Java peut être fait en C# avec Xamarin et Visual Studio.



Xamarin.iOS compile votre application en binaire ARM pour l'App Store d'Apple.



Xamarin.Android utilise un système Just In Time lors de l'exécution.

Il n'y a pas de compromis sur les performances.  
Les applications Xamarin ont une apparence native parce qu'elles sont natives.

# Pourquoi C# ?

# C# is Awesome

- LINQ Support

```
from p in Table<Person> ()  
where p.ID == id  
select p;
```

- XML support

```
var doc = XDocument.Load(url);  
foreach(var item in doc.Root.Elements()) {  
    var text = item.Value;  
}
```

- Event Handling & Delegates

```
button.TouchUpInside += (s, o) => {  
    message.Text = "Hello!";  
};
```

# C# is Awesome - JSON

- Json.NET permet de simplifier la conversion entre vos données JSON et vos objets .NET.

```
public class Person
{
    public string Name { get; set; }
    public DateTime Birthday { get; set; }
}
var person = new Person { Name = "Bob", Birthday = new DateTime (1987, 2, 2) };
var output = Newtonsoft.Json.JsonConvert.SerializeObject (person);

person = Newtonsoft.Json.JsonConvert.DeserializeObject<Person> (output);
Console.WriteLine ("{0} - {1}", person.Name, person.Birthday);
```

# La différence – Classes et Méthodes

## Objective-C

```
// Objective-C
@interface Person : NSObject
@property (strong, nonatomic) NSString *name;
@end

@implementation Person
- (id)initWithName:(NSString *)name {
    self = [super init];
    if (self) {
        self.name = name;
    }
    return self;
}

+ (NSArray *)getNames {
    NSArray *people = @[
        [[Person alloc] initWithName:@"David"],
        [[Person alloc] initWithName:@"Vinicius"],
        [[Person alloc] initWithName:@"Serena"],
    ];
    NSMutableArray *names = [NSMutableArray array];
    [people enumerateObjectsUsingBlock:^(Person *person,
                                         NSUInteger idx,
                                         BOOL *stop) {
        [names addObject:person.name];
    }];
    return names;
}
@end
```

## C# avec Xamarin

```
// C# with Xamarin
class Person : NSObject {
    public string Name { get; set; }

    public static string[] GetNames() {
        var people = new[] {
            new Person { Name="David" },
            new Person { Name="Vinicius" },
            new Person { Name="Serena" },
        };
        return people.Select(person => person.Name).ToArray();
    }
}
```

# La différence – Android ItemClick

Java

```
listView.setOnItemClickListener(new OnItemClickListener() {  
    @Override  
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
  
        // Value of item  
        String itemValue = (String) listView.getItemAtPosition(position);  
  
        // Show Toast  
        Toast.makeText(getApplicationContext(),"Position :" + position + " ListItem : "  
                + itemValue , Toast.LENGTH_LONG).show();  
    }  
});
```

C# avec Xamarin

```
listView.ItemClick += (sender, args) => {  
  
    // Value of item  
    var itemValue = (string)listView.GetItemAtPosition(args.Position);  
  
    //Show Toast  
    Toast.MakeText(this, string.Format("Postition: {0} ListItem: {1}",  
        args.Position, itemValue), ToastLength.Long).Show();  
};
```

C# & Async avec Xamarin

```
listView.ItemClick += async (sender, args) => {  
  
    // Value of item  
    var itemValue = (string)listView.GetItemAtPosition(args.Position);  
  
    //Show Toast  
    Toast.MakeText(this, string.Format("Postition: {0} ListItem: {1}",  
        args.Position, itemValue), ToastLength.Long).Show();  
};
```

# Simplification des API avec Async/Await

## Objective-C

```
[UIView animateWithDuration:0.2
    animations:^{
        view.alpha = 0.0;
    }
    completion:^(BOOL finished){ [view removeFromSuperview]; }];
}
```

## C# avec Xamarin

```
//Animate alpha to 0 asynchronously.
//await animation and then remove from superview
bool success = await UIView.AnimateAsync(2, () => { UIView.Alpha = 0;});
view.RemoveFromSuperview();
```

# Async/Await

Les mots-clés Async et Await dans C # 5.0 maintenant disponibles pour les développeurs Xamarin rendent la programmation asynchrone incroyablement agréable. Vous finissez par un code beaucoup plus linéaire et beaucoup plus facile à comprendre.

```
public async Task ExecuteGetPodcastsCommand()
{
    if (IsBusy)
        return;

    try
    {
        IsBusy = true;
        var client = new HttpClient();
        // Request from server podcast xml
        var podcastString = await client.GetStringAsync(PodcastUrl);

        // Parse Xml into data model and load into list
        var casts = await ParseXml(podcastString);

        foreach (var cast in casts)
        {
            Podcasts.Add(cast);
            FilteredPodcasts.Add(cast);
        }
    }
}
```

Écrivez du code simple & maintenable

# Asynchronisme

- Méthode asynchrone => Retourne Task ou Task<T>

- Exemple :  
`Task<IFile> File.OpenFileAsync(string page)`

```
public async Task ReadData()
{
    IFile file = await File.OpenFileAsync("hello.txt");
    file.ReadAllText();
}
```

- Async permet de déclarer une méthode comme asynchrone (le wrapping du retour en Task sera fait automatiquement)
- Await permet de forcer l'attente de la fin d'une Task et de récupérer le résultat

# Asynchronisme

- Wrapping automatique en Task

```
public async Task<int> Answer()
{
    await Task.Delay(1000);
    return 42;
}
```

- Attention à bien utiliser await sur les méthode asynchrone si vous avez besoin de leur résultat :

```
public async Task ReadData()
{
    IFile file = await File.OpenFileAsync("hello.txt");
    this.Data = file.ReadAllText();
}

public void DisplayContent()
{
    ReadData();
    Console.WriteLine(this.Data);
}
```

# Asynchronisme

- Comment utiliser une méthode asynchrone dans une méthode qu'on ne veux pas asynchrone.

```
public async Task ReadData()
{
    IFile file = await File.OpenFileAsync("hello.txt");
    this.Data = file.ReadAll();
}

public void DisplayContent()
{
    ReadData().ContinueWith(
        (x, state) => Console.WriteLine(this.Data)
        , null);
}
```

```
// A skeleton of a C# program
using System;
namespace YourNamespace
{
    class YourClass
    {
    }

    struct YourStruct
    {
    }

    interface IYourInterface
    {
    }

    delegate int YourDelegate();

    enum YourEnum
    {
    }

    namespace YourNestedNamespace
    {
        struct YourStruct
        {
        }
    }

    class YourMainClass
    {
        static void Main(string[] args)
        {
            //Your program starts here...
        }
    }
}
```



# C#

<b>Paradigm</b>	multi-paradigm; structured, imperative, object-oriented, event-driven, task-driven, functional, generic, reflective, concurrent
<b>Family</b>	C
<b>Designed by</b>	Microsoft
<b>Developer</b>	Microsoft
<b>First appeared</b>	2000; 17 years ago <sup>[1]</sup>
<b>Stable release</b>	6.0 <sup>[2]</sup> / 2015; 2 years ago
<b>Preview release</b>	7 / 2016; 1 year ago <sup>[3]</sup>
<b>Typing discipline</b>	static, dynamic, <sup>[4]</sup> strong, safe, nominative, partially inferred
<b>Platform</b>	Common Language Infrastructure
<b>License</b>	CLR is proprietary, Mono compiler is dual GPLv3, MIT/X11 and libraries are LGPLv2, DotGNU is dual GPL and LGPLv2
<b>Filename extensions</b>	.cs
<b>Website</b>	<a href="http://www.visualstudio.com">www.visualstudio.com</a>
<b>Major implementations</b>	
Visual C#, .NET Framework, Mono, DotGNU	
<b>Dialects</b>	
Cw, Spec#, Polyphonic C#, Enhanced C# <sup>[5]</sup>	
<b>Influenced by</b>	
C++, <sup>[6]</sup> Eiffel, Java, <sup>[5]</sup> Modula-3, Object Pascal, <sup>[8]</sup> ML, VB, Icon, Haskell, Rust, J#, Cw, F# <sup>[note 1]</sup>	
<b>Influenced</b>	
Chapel, <sup>[7]</sup> D, J#, Dart, <sup>[8]</sup> F#, Hack, Java, <sup>[9]</sup> <sup>[10]</sup> Kotlin, Monkey, Nemerle, Oxygene, Rust, Swift, <sup>[11]</sup> Vala	
<b>C Sharp Programming at Wikibooks</b>	

# General Structure of C# program

- C# programs can contain one or more files. Each file can have zero or more namespaces. A namespace can contain types such as classes, structures, interfaces, enumerations, and delegates, in addition to other namespaces.
- The following C# program skeleton contains all of these elements.

```
// A skeleton of a C# program
using System;
namespace YourNamespace
{
    class YourClass
    {

    }

    struct YourStruct
    {

    }

    interface IYourInterface
    {

    }

    delegate int YourDelegate();

    enum YourEnum
    {

    }

    namespace YourNestedNamespace
    {
        struct YourStruct
        {

        }
    }

    class YourMainClass
    {
        static void Main(string[] args)
        {
            //Your program starts here...
        }
    }
}
```

# C# - Basic Syntax – Identifier

- Name of an element in the [code](#). There are certain standard [naming conventions](#) to follow when selecting names for elements:
  - An identifier can:
    - start with an underscore: `_`
    - contain an underscore: `_`
    - contain a numeral: `0123456789`
    - contain both [upper case and lower case](#) Unicode letters. Case is sensitive (BOX is different from box).
  - An identifier cannot:
    - start with a numeral
    - start with a symbol, unless it is a keyword (check [Keywords](#))
    - have more than 511 [chars](#)
    - contain @ sign in between or at the end

# C# - Basic Syntax – Keywords

Keywords are predefined reserved words with special syntactic meaning. The language has two types of keyword — contextual and reserved. The reserved keywords such as false or byte may only be used as keywords. The contextual keywords such as where or from are only treated as keywords in certain situations.[If an identifier is needed which would be the same as a reserved keyword, it may be prefixed by the @ character to distinguish it. This facilitates reuse of .NET code written in other languages.

C# keywords, reserved words			
abstract	as	base	bool
break	by <sup>2</sup>	byte	case
catch	char	checked	class
const	continue	decimal	default
delegate	do	double	descending <sup>2</sup>
explicit	event	extern	else
enum	false	finally	fixed
float	for	foreach	from <sup>2</sup>
goto	group <sup>2</sup>	if	implicit
in	int	interface	internal
into <sup>2</sup>	is	lock	long
new	null	namespace	object
operator	out	override	orderby <sup>2</sup>
params	private	protected	public
readonly	ref	return	switch
struct	sbyte	sealed	short
sizeof	stackalloc	static	string
select <sup>2</sup>	this	throw	true
try	typeof	uint	ulong
unchecked	unsafe	ushort	using
var <sup>2</sup>	virtual	volatile	void
while	where <sup>1[3]2</sup>	yield <sup>1</sup>	

<sup>1,2</sup> These are not actually keywords, thus (unlike actual keywords) it is possible to define variables and types using these names, but they act like keywords in certain new language constructs introduced in C# 2.0<sup>(1)</sup> and 3.0<sup>(2)</sup>.

# C# - Basic Syntax – Value Type

To get the exact size of a type or a variable on a particular platform, you can use the `sizeof` method. The expression `sizeof(type)` yields the storage size of the object or type in bytes.

Type	Represents	Range	Default Value
bool	Boolean value	True or False	False
byte	8-bit unsigned integer	0 to 255	0
char	16-bit Unicode character	U +0000 to U +ffff	'\0'
decimal	128-bit precise decimal values with 28-29 significant digits	(-7.9 x 10 <sup>28</sup> to 7.9 x 10 <sup>28</sup> ) / 10 <sup>0</sup> to 28	0.0M
double	64-bit double-precision floating point type	(+/-)5.0 x 10 <sup>-324</sup> to (+/-)1.7 x 10 <sup>308</sup>	0.0D
float	32-bit single-precision floating point type	-3.4 x 10 <sup>38</sup> to + 3.4 x 10 <sup>38</sup>	0.0F
int	32-bit signed integer type	-2,147,483,648 to 2,147,483,647	0
long	64-bit signed integer type	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0L
sbyte	8-bit signed integer type	-128 to 127	0
short	16-bit signed integer type	-32,768 to 32,767	0
uint	32-bit unsigned integer type	0 to 4,294,967,295	0
ulong	64-bit unsigned integer type	0 to 18,446,744,073,709,551,615	0
ushort	16-bit unsigned integer type	0 to 65,535	0

# C# - Basic Syntax - Implicitly Typed Local Variables

- Beginning in Visual C# 3.0, variables that are declared at method scope can have an implicit type var. An implicitly typed local variable is strongly typed just as if you had declared the type yourself, but the compiler determines the type. The following two declarations of i are functionally equivalent:
- Local variables can be given `var i = 10; // implicitly typed` instead of an explicit type. The var keyword instructs the compiler to infer the variable from the expression on the right side of the initialization statement. The inferred type may be a built-in type, an anonymous type, a user-defined type, or a type defined in the .NET Framework class library.

# C# - Basic Syntax - Implicitly Typed Local Variables

```
// i is compiled as an int  
var i = 5;  
  
// s is compiled as a string  
var s = "Hello";  
  
// a is compiled as int[]  
var a = new[] { 0, 1, 2 };  
  
// expr is compiled as IEnumerable<Customer>  
// or perhaps IQueryable<Customer>  
var expr =  
    from c in customers  
    where c.City == "London"  
    select c;  
  
// anon is compiled as an anonymous type  
var anon = new { Name = "Terry", Age = 34 };  
  
// list is compiled as List<int>  
var list = new List<int>();
```

## Remarks

The following restrictions apply to implicitly-typed variable declarations:

- var can only be used when a local variable is declared and initialized in the same statement; the variable cannot be initialized to null, or to a method group or an anonymous function.
- var cannot be used on fields at class scope.
- Variables declared by using var cannot be used in the initialization expression. In other words, this expression is legal: int i = (i = 20); but this expression produces a compile-time error: var i = (i = 20);
- Multiple implicitly-typed variables cannot be initialized in the same statement.
- If a type named var is in scope, then the var keyword will resolve to that type name and will not be treated as part of an implicitly typed local variable declaration.

```
for(var x = 1; x < 10; x++)
```

```
foreach(var item in list){...}
```

# C# - Basic Syntax – Literals

Integers	
hexadecimal	0xF5, 0x[0..9, A..F, a..f]+
decimal	245, [0..9]+
Floating-point values	
float	23.5F, 23.5f; 1.72E3F, 1.72E3f, 1.72e3F, 1.72e3f
double	23.5, 23.5D, 23.5d; 1.72E3, 1.72E3D, ...
Dates	
DateTime	DateTime.Now; // (Date and time as and when it's assigned)
Characters	
char	'a', 'Z', '\u0231'
Strings	
String	"Hello, world" "C:\\Windows\\", @"C:\\Windows\\"
Characters escapes in strings	
Unicode character	\u followed by the hexadecimal unicode code point
Null character <sup>1</sup>	\0
Tab	\t
Backspace	\b
Carriage return	\r
Form feed	\f
Backslash	\\\
Single quote	\'
Double quote	\"
Line feed	\n

<sup>1</sup> Strings in C# are not null terminated

# C# - Basic Syntax – Operators

Operator category	Operators
Arithmetic	<code>+, -, *, /, %</code>
Logical (boolean and bitwise)	<code>&amp;,  , ^, !, ~, &amp;&amp;,   , true, false</code>
String concatenation	<code>+</code>
Increment, decrement	<code>++, --</code>
Shift	<code>&lt;&lt;, &gt;&gt;</code>
Relational (conditional)	<code>==, !=, &lt;, &gt;, &lt;=, &gt;=</code>
Assignment	<code>=, +=, -=, *=, /=, %=, &amp;=,  =, ^=, &lt;&lt;=, &gt;&gt;=</code>
Member access	<code>.</code>
Indexing	<code>[]</code>
Cast	<code>()</code>
Conditional	<code>?:</code>
Delegate concatenation and removal	<code>+, -</code>
Object creation	<code>new</code>
Type information	<code>as, is, sizeof, typeof</code>
Overflow exception control	<code>checked, unchecked</code>
Indirection and Address	<code>*, -&gt;, [], &amp;</code>
Coalesce	<code>??</code>
Lambda expression	<code>=&gt;</code>

# C# - Basic Syntax – Variables

Variables are identifiers associated with values. They are declared by writing the variable's type and name, and are optionally initialized in the same statement.

- Defining Variables: syntax for variable definition in C# is: **<data\_type>  
<variable\_list>;**

Exemples:

```
int i, j, k;  
char c, ch;  
float f, salary;  
double d;
```

- Initializing Variables : variables are initialized (assigned a value) with an equal sign followed by a constant expression. The general form of initialization is: **variable\_name = value;**
- Variables can be initialized in their declaration. The initializer consists of an equal sign followed by a constant expression as: **<data\_type>  
<variable\_name> = value;**

# Example uses various types of variables

The screenshot shows the Visual Studio IDE interface with the following details:

- Project Structure:** The solution contains three files: `Program.cs`, `AssemblyInfo.cs`, and `MyFirstConsoleApplication.cs`.
- Code Editor:** The `Program.cs` file is open, showing the following C# code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MyFirstConsoleApplication
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             short a;
14             int b;
15             double c;
16
17             /* actual initialization */
18             a = 10;
19             b = 20;
20             c = a + b;
21             Console.WriteLine("a = {0}, b = {1}, c = {2}", a, b, c);
22             Console.ReadLine();
23         }
24     }
25 }
```
- Output Window:** The bottom window displays the console output:

```
a = 10, b = 20, c = 30
```

# Constants

- Constants are immutable values which are known at compile time and do not change for the life of the program. Constants are declared with the const modifier.

```
const int months = 12;
const int weeks = 52;
const int days = 365;

const double daysPerWeek = (double) days / (double) weeks;
const double daysPerMonth = (double) days / (double) months;
```

# C# - Basic Syntax – Constants

- When declaring a local variable or a field with the **const** keyword as a prefix the value must be given when it is declared. After that it is locked and cannot change. They can either be declared in the context as a field or a local variable. Constants are implicitly static.

```
const double PI = 3.14;
```

```
class Foo {  
    const double X = 3;  
    Foo() {  
        const int Y = 2;  
    }  
}
```

# C# - Basic Syntax – Comments

- Comments are used for explaining code. Compilers ignore the comment entries. The multiline comments in C# programs start with /\* and terminates with the characters \*/ as shown below:

- Single-line comments are symbol.

```
/* This program demonstrates  
The basic syntax of C# programming  
Language */
```

```
}//end class Rectangle
```

# C# - Basic Syntax – enum

- The enum keyword is used to declare an enumeration, a distinct type that consists of a set of named constants called the enumerator list.
- Usually it is best to define an enum directly within a namespace so that all classes in the namespace can access it with equal convenience. However, an enum can also be nested within a class or struct.

By default, the first enumerator has the value 0, and the value of each successive enumerator is increased by 1. For example, in the following enumeration, Sat is 0, Sun is 1, Mon is 2, and so forth.

```
public class EnumTest
{
    enum Days { Sun, Mon, Tue, Wed, Thu, Fri, Sat };

    static void Main()
    {
        int x = (int)Days.Sun;
        int y = (int)Days.Fri;
        Console.WriteLine("Sun = {0}", x);
        Console.WriteLine("Fri = {0}", y);
    }
}
/* Output:
   Sun = 0
   Fri = 5
```

# Reading value from user

- **Console** class in the **System** namespace provides a function **ReadLine()** for accepting input from the user and store it into a variable.

For example,

```
int num;  
num = Convert.ToInt32(Console.ReadLine());
```

The function **Convert.ToInt32()** converts the data entered by the user to int data type, because **Console.ReadLine()** accepts the data in string format.

# C# - Basic Syntax – Conditional structures

- if statement is entered when the given condition is true. Single-line case statements do not require block braces although it is mostly preferred by convention.
- Simple one-line statement:
- Multi-line with else-block (without any block braces)  
`if (i == 3) ... ;`

```
if (i == 2)
    ...
else
    ...
```

# C# - Basic Syntax – Conditional structures

- The switch construct serves as a filter for different values. Each value leads to a "case". It is not allowed to fall through case sections and therefore the keyword break is typically used to end a case. An unconditional return in a case section can also be used to end a case.

```
switch (ch)
{
    case 'A':
        statement;
        ...
        break;
    case 'B':
        statement;
        break;
    case 'C': // A switch section can have multiple case Labels.
    case 'D':
        ...
        break;
    default:
        ...
        break;
}
```

```
Console.WriteLine("Donner un nombre:");
int nb = Int32.Parse(Console.ReadLine());
switch (nb)
{
    case 1: Console.WriteLine("Lundi"); break;
    case 2: Console.WriteLine("Mardi"); break;
    case 3: Console.WriteLine("Mercredi"); break;
    default: Console.WriteLine("Autrement");
    break;
}
```

# C# - Basic Syntax

Iteration statements are statements that are repeatedly executed when a given condition is evaluated as true.

- while loop

```
while (i == true)
{
    ...
}
```

```
int s=0;int i=0;
while (i<10){
    s+=i;
    i++;
}
Console.WriteLine("Somme="+s);
```

- do ... while loop

```
do
{
    ...
}
while (i == true);
```

```
int s=0;int i=0;
do{
    s+=i;
    i++;
}while (i<10);
Console.WriteLine("Somme="+s);
```

- for loop

```
for (int i = 0; i < 10; i++)
{
    ...
}
```

```
for (int i = 2; i < 10;i++) {
    Console.WriteLine
    ("I="+i);
}
```

- foreach loop

```
foreach (int i in intList)
{
    ...
}
```

```
string[] amis = { "A", "B", "C", "D" };
foreach (string nom in amis) {
    Console.WriteLine(nom);
}
```

# Arrays

```
int[] numbers = new int[2];
numbers[0] = 2;
numbers[1] = 5;
int x = numbers[0];
```

- An array type is a reference type that refers to a space containing one or more elements of a certain type. All array types derive from a common base class, System.Array. Each element is referenced by its index just like in C++ and Java.
- Initializers: Array initializers provide convenient syntax for initialization of arrays.

```
// Long syntax
int[] numbers = new int[5]{ 20, 1, 42, 15, 34 };
// Short syntax
int[] numbers2 = { 20, 1, 42, 15, 34 };
// Inferred syntax var numbers3 = new[] { 20, 1, 42, 15, 34 };
```

# Arrays

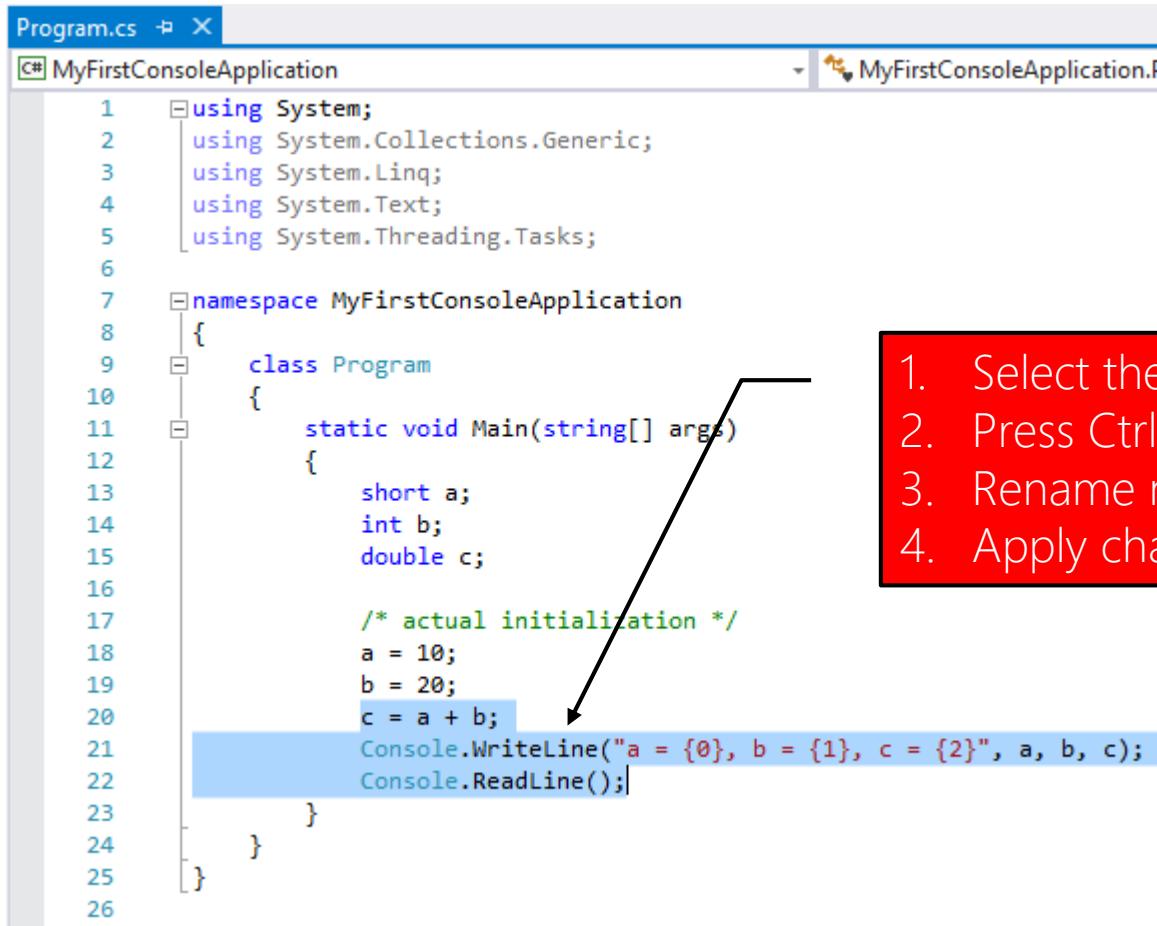
- Arrays can have more than one dimension, for example 2 dimensions to represent a grid.

```
int[,] numbers = new int[3, 3];  
  
numbers[1,2] = 2;  
  
int[,] numbers2 = new int[3, 3] { {2, 3, 2}, {1, 2, 6}, {2, 4, 5} };
```

# Methods

- Like in C and C++ there are functions that group reusable code. The main difference is that functions, just like in Java, have to reside inside of a class. A function is therefore called a *method*. A method has a return value, a name and usually some parameters initialized when it is called with some arguments. It can either belong to an instance of a class or be a static member.

# Methods



```
Program.cs  X
MyFirstConsoleApplication
MyFirstConsoleApplication.P

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MyFirstConsoleApplication
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             short a;
14             int b;
15             double c;
16
17             /* actual initialization */
18             a = 10;
19             b = 20;
20             c = a + b;
21             Console.WriteLine("a = {0}, b = {1}, c = {2}", a, b, c);
22             Console.ReadLine();
23         }
24     }
25 }
26
```

1. Select these three lines
2. Press Ctrl+(R and after M)
3. Rename new method to « somme »
4. Apply change

# Methods

The screenshot shows a C# code editor with the file `Program.cs` open. The code defines a `Program` class with a `Main` method. Inside `Main`, there is a call to a `somme` method. The `somme` method is defined as a private static double method that takes two parameters (`a` and `b`) and returns their sum. The `somme` method also outputs the values of `a`, `b`, and `c` to the console.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MyFirstConsoleApplication
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             short a;
14             int b;
15             double c;
16
17             /* actual initialization */
18             a = 10;
19             b = 20;
20             c = somme(a, b);
21             Console.ReadLine();
22         }
23
24         private static double somme(short a, int b)
25         {
26             double c = a + b;
27             Console.WriteLine("a = {0}, b = {1}, c = {2}", a, b, c);
28             return c;
29         }
30     }
31 }
```

# Passing Parameter to Method

The screenshot shows the Visual Studio IDE with a C# project named "MyFirstConsoleApplication". The code editor displays the file "Program.cs" with the following content:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MyFirstConsoleApplication
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             int age = 20;
14             ChangeInt(age);
15             Console.WriteLine("Effective Parameter age=" + age);
16             Console.ReadLine();
17         }
18         private static void ChangeInt(int a)
19         {
20             a = 30;
21             Console.WriteLine("Formal Parameter a=" + a);
22         }
23     }
24 }
25
26
```

The output window at the bottom shows the following text:

```
Formal Parameter a=30
Effective Parameter age=20
```

# Passing Parameter to Method

- **Parameter passing by value:** The previous example shows that the parameters of a function are by default passed by value, ie the value of the effective parameter is copied into the corresponding formal parameter. On two separate entities. If the function changes the formal parameter, the effective parameter is not changed.

# Passing Parameter to Method

- **Passing parameters by reference:** In a passage by reference, the effective parameter and the formal parameter are one and the same entity. If the function changes the formal parameter, the effective parameter is also changed. In C #, they must both be preceded by the keyword ref

# Passing Parameter to Method

The screenshot shows a Microsoft Visual Studio IDE window. The title bar says "Program.cs" and "MyFirstConsoleApplication". The code editor displays the following C# code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace MyFirstConsoleApplication
8  {
9      class Program
10     {
11         static void Main(string[] args)
12         {
13             int age = 20;
14             ChangeInt(ref age);
15             Console.WriteLine("Effective Parameter age=" + age);
16             Console.ReadLine();
17         }
18         private static void ChangeInt(ref int a)
19         {
20             a = 30;
21             Console.WriteLine("Formal Parameter a=" + a);
22         }
23     }
24 }
25
26 file:///C:/Users/toh/documents/visual studio 2015/Projects/MyFirstConsoleApplication/MyFirstConsoleApplication/bin/Debug/MyFirstConsoleApplicat... — X
```

The output window at the bottom shows the results of the program's execution:

```
Formal Parameter a=30
Effective Parameter age=30
```

# Strings

- A string is an object of type String whose value is text. Internally, the text is stored as a sequential read-only collection of Char objects.
- `string` vs. `System.String`: `string` keyword is an alias for `String`. Therefore, `String` and `string` are equivalent, and you can use whichever naming convention you prefer.

# Declaring and Initializing Strings

```
// Declare without initializing.  
string message1;  
  
// Initialize to null.  
string message2 = null;  
  
// Initialize as an empty string.  
// Use the Empty constant instead of the literal "".  
string message3 = System.String.Empty;  
  
// Initialize with a regular string literal.  
string oldPath = "c:\\Program Files\\Microsoft Visual Studio 8.0";  
  
// Initialize with a verbatim string literal.  
string newPath = @"c:\\Program Files\\Microsoft Visual Studio 9.0";  
  
// Use System.String if you prefer.  
System.String greeting = "Hello World!";  
  
// In local variables (i.e. within a method body)  
// you can use implicit typing.  
var temp = "I'm still a strongly-typed System.String!";  
  
// Use a const string to prevent 'message4' from  
// being used to store another string value.  
const string message4 = "You can't get rid of me!";  
  
// Use the String constructor only when creating  
// a string from a char*, char[], or sbyte*. See  
// System.String documentation for details.  
char[] letters = { 'A', 'B', 'C' };  
string alphabet = new string(letters);
```

# String Escape Sequences

Escape sequence	Character name	Unicode encoding
\'	Single quote	0x0027
\"	Double quote	0x0022
\	Backslash	0x005C
\0	Null	0x0000
\a	Alert	0x0007
\b	Backspace	0x0008
\f	Form feed	0x000C
\n	New line	0x000A
\r	Carriage return	0x000D
\t	Horizontal tab	0x0009
\U	Unicode escape sequence for surrogate pairs.	\Unnnnnnnn
\u	Unicode escape sequence	\u0041 = "A"
\v	Vertical tab	0x000B
\x	Unicode escape sequence similar to "\u" except with variable length.	\x0041 = "A"

# Format Strings

```
class FormatString
{
    static void Main()
    {
        // Get user input.
        System.Console.WriteLine("Enter a number");
        string input = System.Console.ReadLine();

        // Convert the input string to an int.
        int j;
        System.Int32.TryParse(input, out j);

        // Write a different string each iteration.
        string s;
        for (int i = 0; i < 10; i++)
        {
            // A simple format string with no alignment formatting.
            s = System.String.Format("{0} times {1} = {2}", i, j, (i * j));
            System.Console.WriteLine(s);
        }

        //Keep the console window open in debug mode.
        System.Console.ReadKey();
    }
}
```

# Substrings

- A substring is any sequence of characters that is contained in a string. Use the [Substring](#) method to create a new string from a part of the original string.
- You can search for one or more occurrences of a substring by using the [IndexOf](#) method.
- Use the [Replace](#) method to replace all occurrences of a specified substring with a new string. Like the [Substring](#) method, [Replace](#) actually returns a new string and does not modify the original string.

```
string s3 = "Visual C# Express";
System.Console.WriteLine(s3.Substring(7, 2));
// Output: "C#"

System.Console.WriteLine(s3.Replace("C#", "Basic"));
// Output: "Visual Basic Express"

// Index values are zero-based
int index = s3.IndexOf("C");
// index = 7
```

# Accessing Individual Characters

```
string s5 = "Printing backwards";

for (int i = 0; i < s5.Length; i++)
{
    System.Console.WriteLine(s5[s5.Length - i - 1]);
}
// Output: "sdrawkcab gnitnirP"
```

# Exercices

- Create a c# program that asks the user his / her name, first name, date of birth, place of birth and then displays each of these information on a line.
- Write a c# program that instructs a user to enter the coefficients of a first degree equation ( $ax + b = 0$ ) and calculate and display the solution.
- Write a c# program that instructs a user to enter a time T in seconds, then converts it into hours, minutes, and seconds and displays the corresponding values.

# Exercices

- Write a c# program to know whether a user entered a leap year or not. Reminder: a leap year is divisible by 400 or by 4 but not divisible by 100.
  - Write a c# program to determine the next day's date based on the day, month and year of a given date entered by the user.
  - Using a double loop, make a cone to draw the top of the fir on 10 lines.

# Exercices

- Write the program C # which calculates the discriminant DELTA of a trinomial of the second degree  $AX^2 + BX + C$  and which, according to its sign, calculates solutions. The three coefficients A, B and C will be entered on the keyboard before processing.

# Exercices

- Write a C# program that print a table of 10 lines and 10 columns with random character. You have to set the same random character for all elements that are multiple of 9.

The screenshot shows a terminal window titled "C:\Program Files\dotnet\dotnet.exe". The window displays a character mapping table and a user interaction prompt. The table maps numbers from 0 to 99 to characters. A dashed line separates the table from the user input prompt. The user has typed "g" and the program has responded with "- The caracter to which you are thinking is g -". A final prompt at the bottom asks the user to press any key to continue.

***** Telepathy *****									
1. Choose a Number between 0 and 99									
2. Make difference between that number and the two one that compose it									
3. Look at the table and thougt deeply to the caracter corresponding to the obtain number									
4. Press any Key and I will say you the letter to which you are thinking !!!									
0 g	1 k	2	3 TÉ	4 TÁ	5 Tf	6 P	7 Të	8 k	9 g
10 u	11 f	12 P	13 [	14 Tè	15 TÆ	16 T	17 D	18 g	19 J
20 ]	21 TÇ	22 Tò	23 w	24 Tö	25	26 [	27 g	28 Tå	29 D
30 }	31 Tá	32 q	33 ]	34 TÆ	35 ~	36 g	37 J	38 T£	39 J
40 w	41 Tù	42 q	43 ~	44 L	45 g	46 C	47 n	48 s	49 i
50 H	51 ^	52 TÇ	53 m	54 g	55 o	56 {	57 F	58 Tí	59 d
60 Tà	61 Tà	62 D	63 g	64 y	65 y	66 TÜ	67 Tù	68 n	69 M
70 TÄ	71 J	72 g	73 TÖ	74 x	75 j	76 O	77 u	78 l	79 }
80 z	81 g	82 TÍ	83 ^	84 Tí	85 _	86 a	87 E	88 Tö	89 c
90 g	91 TÚ	92 N	93 y	94 j	95 Tù	96 Tö	97 J	98 Tî	99 g

- The caracter to which you are thinking is g -

Press any key to conitnue...

# Webography

- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://benjii.me/>
- [http://www.w3schools.com/asp/webpages\\_razor.asp](http://www.w3schools.com/asp/webpages_razor.asp)
- <https://blogs.msdn.microsoft.com/webdev/>
- [https://www.tutorialspoint.com/asp.net\\_core/](https://www.tutorialspoint.com/asp.net_core/)
- [https://www.youtube.com/results?search\\_query=ASP.NET+Core+2017](https://www.youtube.com/results?search_query=ASP.NET+Core+2017)

# C#

**Paradigm** multi-paradigm; structured, imperative, object-oriented, event-driven, task-driven, functional, generic, reflective, concurrent

**Family** C

**Designed by** Microsoft

**Developer** Microsoft

**First appeared** 2000; 17 years ago<sup>[1]</sup>

**Stable release** 6.0<sup>[2]</sup> / 2015; 2 years ago

**Preview release** 7 / 2016; 1 year ago<sup>[3]</sup>

**Typing discipline** static, dynamic,<sup>[4]</sup> strong, safe, nominative, partially inferred

**Platform** Common Language Infrastructure

**License** CLR is proprietary, Mono compiler is dual GPLv3, MIT/X11 and libraries are LGPLv2, DotGNU is dual GPL and LGPLv2

**Filename extensions** .cs

**Website** [www.visualstudio.com](http://www.visualstudio.com)<sup>[5]</sup>

#### Major implementations

Visual C#, .NET Framework, Mono, DotGNU

#### Dialects

Cw, Spec#, Polyphonic C#, Enhanced C#<sup>[6]</sup>

#### Influenced by

C++,<sup>[5]</sup> Eiffel, Java,<sup>[5]</sup> Modula-3, Object Pascal,<sup>[8]</sup> ML, VB, Icon, Haskell, Rust, J#, Cw, F#<sup>[note 1]</sup>

#### Influenced

Chapel,<sup>[7]</sup> D, J#, Dart,<sup>[8]</sup> F#, Hack, Java,<sup>[9][10]</sup> Kotlin, Monkey, Nemerle, Oxygene, Rust, Swift,<sup>[11]</sup> Vala

 C Sharp Programming at Wikibooks

```
namespace ProgrammingGuide
{
    // Class definition.
    public class MyCustomClass
    {
        // Class members:
        // Property.
        public int Number { get; set; }

        // Method.
        public int Multiply(int num)
        {
            return num * Number;
        }

        // Instance Constructor.
        public MyCustomClass()
        {
            Number = 0;
        }
    }

    // Another class definition. This one contains
    // the Main method, the entry point for the program.
    class Program
    {
        static void Main(string[] args)
        {
            // Create an object of type MyCustomClass.
            MyCustomClass myClass = new MyCustomClass();

            // Set the value of a public property.
            myClass.Number = 27;

            // Call a public method.
            int result = myClass.Multiply(4);

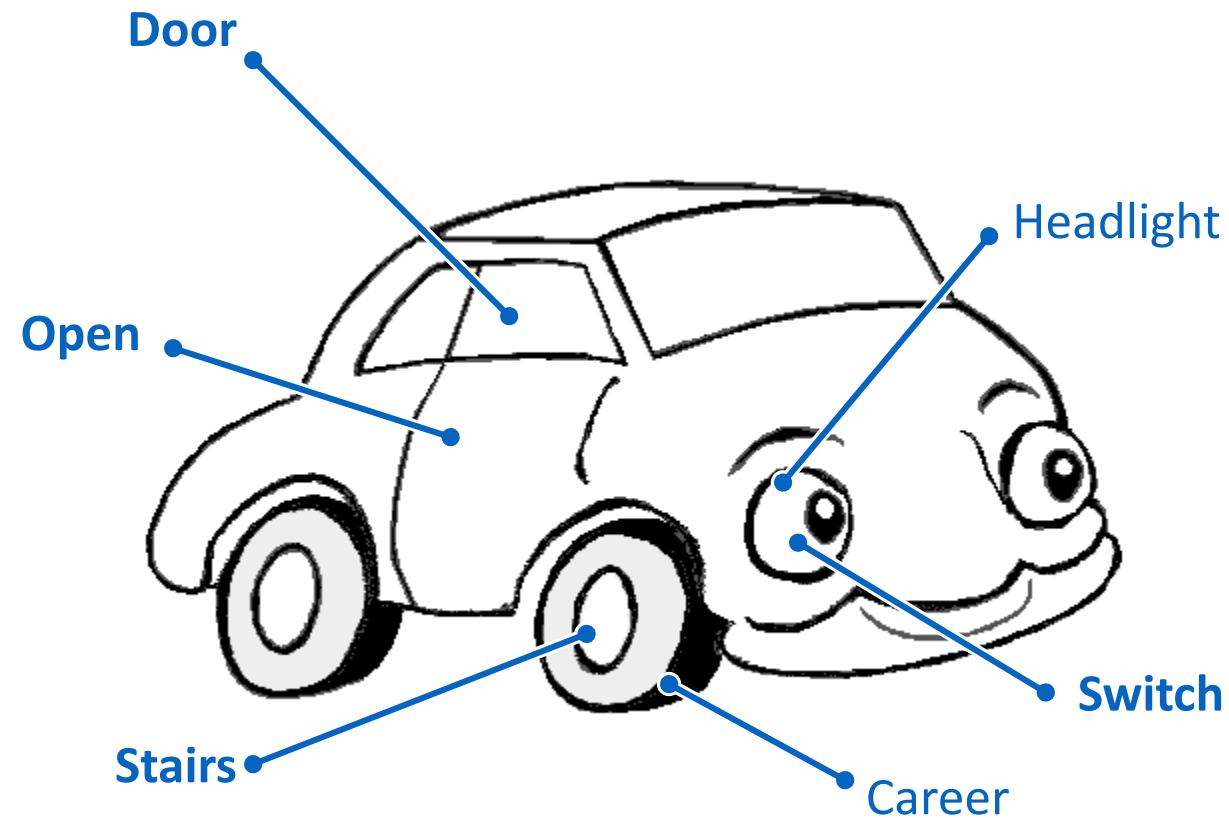
        }
    }
}
```

# C# | Object Oriented Programming

# Encapsulation

- Gather in the same structure:
  - Member variables (data)
  - Methods (functions)
- Ensure the integrity of the data by:
  - Only leaving visible what must be actually used (visibility)
  - Accessing data only by methods
- Class = abstract description of an object
- Instantiate a class = create an object on its model (thanks to the constructor)
- Property = member variable accessible by a getter and / or setter

# Classes and objects



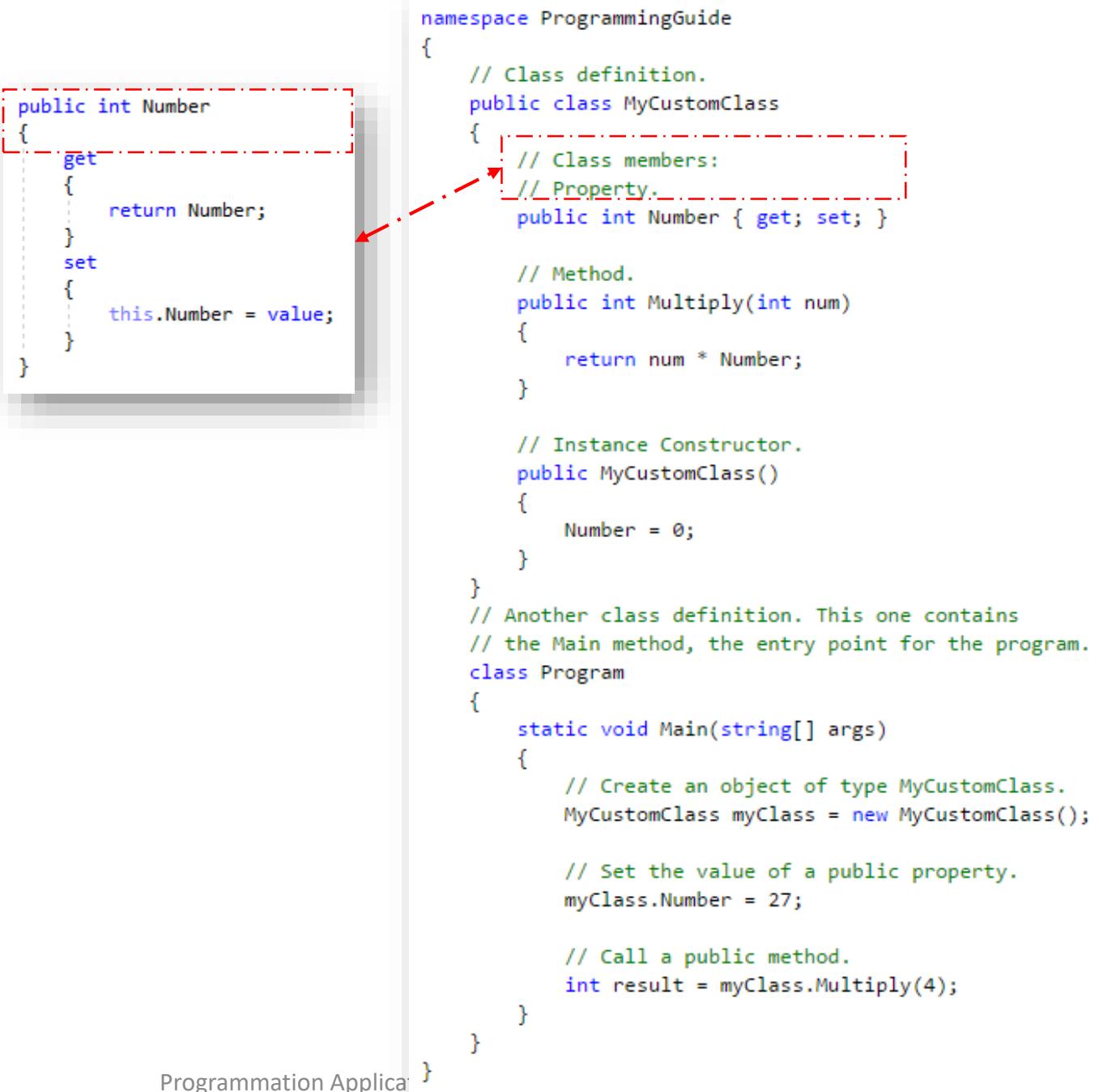
# Classes and Structs

- Classes and structs are two of the basic constructs of the common type system in the .NET Framework.
- Each is essentially a data structure that encapsulates a set of data and behaviors that belong together as a logical unit.
- The data and behaviors are the *members* of the class or struct, and they include its methods, properties, and events

# Classes and Structs

- A class or struct declaration is like a blueprint that is used to create instances or objects at run time.
- A class is a reference type. When an object of the class is created, the variable to which the object is assigned holds only a reference to that memory.
- A struct is a value type. When a struct is created, the variable to which the struct is assigned holds the struct's actual data.

# Classes and Structs



```
namespace ProgrammingGuide
{
    // Class definition.
    public class MyCustomClass
    {
        // Class members:
        // Property.
        public int Number { get; set; }

        // Method.
        public int Multiply(int num)
        {
            return num * Number;
        }

        // Instance Constructor.
        public MyCustomClass()
        {
            Number = 0;
        }
    }
    // Another class definition. This one contains
    // the Main method, the entry point for the program.
    class Program
    {
        static void Main(string[] args)
        {
            // Create an object of type MyCustomClass.
            MyCustomClass myClass = new MyCustomClass();

            // Set the value of a public property.
            myClass.Number = 27;

            // Call a public method.
            int result = myClass.Multiply(4);
        }
    }
}
```

# Access Modifiers

- You can use the following access modifiers to specify the accessibility of a type or member when you declare it:
  - **public** :The type or member can be accessed by any other code in the same assembly or another assembly that references it.
  - **private** : The type or member can be accessed only by code in the same class or struct.
  - **protected** :The type or member can be accessed only by code in the same class or struct, or in a class that is derived from that class.
  - **internal** :The type or member can be accessed by any code in the same assembly, but not from another assembly.
  - **protected internal** : The type or member can be accessed by any code in the assembly in which it is declared, or from within a derived class in another assembly.

```
// public class:  
public class Tricycle  
{  
    // protected method:  
    protected void Pedal() { }  
  
    // private field:  
    private int wheels = 3;  
  
    // protected internal property:  
    protected internal int Wheels  
    {  
        get { return wheels; }  
    }  
}
```

# Fields

- A field is a variable of any type that is declared directly in a class or struct. Fields are members of their containing type.

```
1  +using ...
2
3  -namespace MyFirstConsoleApplication
4  {
5      class CalendarEntry
6      {
7          // private field
8          private DateTime date;
9
10         // public field (Generally not recommended.)
11         public string day;
12
13         // Public property exposes date field safely.
14         public DateTime Date
15         {
16             get
17             {
18                 return date;
19             }
20             set
21             {
22                 // Set some reasonable boundaries for likely birth dates.
23                 if (value.Year > 1900 && value.Year <= DateTime.Today.Year)
24                 {
25                     date = value;
26                 }
27                 else
28                     throw new ArgumentOutOfRangeException();
29             }
30
31         }
32
33         // Public method also exposes date field safely.
34         // Example call: birthday.SetDate("1975, 6, 30");
35         public void SetDate(string dateString)
36         {
37             DateTime dt = Convert.ToDateTime(dateString);
38
39             // Set some reasonable boundaries for likely birth dates.
40             if (dt.Year > 1900 && dt.Year <= DateTime.Today.Year)
41             {
42                 date = dt;
43             }
44             else
45                 throw new ArgumentOutOfRangeException();
46         }
47     }
48 }
49 }
```

# Properties

- A property is a member that provides a flexible mechanism to read, write, or compute the value of a private field. Properties can be used as if they are public data members, but they are actually special methods called accessors. This enables data to be accessed easily and still helps promote the safety and flexibility of methods.

```
class TimePeriod
{
    private double seconds;

    public double Hours
    {
        get { return seconds / 3600; }
        set { seconds = value * 3600; }
    }
}

class Program
{
    static void Main()
    {
        TimePeriod t = new TimePeriod();

        // Assigning the Hours property causes the 'set' accessor to be called.
        t.Hours = 24;

        // Evaluating the Hours property causes the 'get' accessor to be called.
        System.Console.WriteLine("Time in hours: " + t.Hours);
    }
}
// Output: Time in hours: 24
```

# Constructors

- Whenever a [class](#) or [struct](#) is created, its constructor is called. A class or struct may have multiple constructors that take different arguments. Constructors enable the programmer to set default values, limit instantiation, and write code that is flexible and easy to read.
- If you do not provide a constructor for your object, C# will create one by default that instantiates the object and sets member variables to the default values as listed in [Default Values Table](#).

# Constructors

- When a class or struct is created, its constructor is called. Constructors have the same name as the class or struct, and they usually initialize the data members of the new object.
- In the following example, a class named Taxi is defined by using a simple constructor. This class is then instantiated with the new operator. The Taxi constructor is invoked by the new operator immediately after memory is allocated for the new object.

```
public class Taxi
{
    public bool isInitialized;
    public Taxi()
    {
        isInitialized = true;
    }
}

class TestTaxi
{
    static void Main()
    {
        Taxi t = new Taxi();
        Console.WriteLine(t.isInitialized);
    }
}
```

# Constructors

- The following example demonstrates a class with two class constructors, one without arguments and one with two arguments.

```
class CoOrds
{
    public int x, y;

    // Default constructor:
    public CoOrds()
    {
        x = 0;
        y = 0;
    }

    // A constructor with two arguments:
    public CoOrds(int x, int y)
    {
        this.x = x;
        this.y = y;
    }

    // Override the ToString method:
    public override string ToString()
    {
        return String.Format("({0},{1})", x, y);
    }
}

class MainClass
{
    static void Main()
    {
        CoOrds p1 = new CoOrds();
        CoOrds p2 = new CoOrds(5, 3);

        // Display the results using the overridden ToString method:
        Console.WriteLine("CoOrds #1 at {0}", p1);
        Console.WriteLine("CoOrds #2 at {0}", p2);
        Console.ReadKey();
    }
}
/* Output:
CoOrds #1 at (0,0)
CoOrds #2 at (5,3)
*/
```

# Object and Collection Initializers

- Object initializers let you assign values to any accessible fields or properties of an object at creation time without having to invoke a constructor followed by lines of assignment statements. The object initializer syntax enables you to specify arguments for a constructor or omit the arguments (and parentheses syntax).

```
class Cat
{
    // Auto-implemented properties.
    public int Age { get; set; }
    public string Name { get; set; }
}
Cat cat = new Cat { Age = 10, Name = "Fluffy" };
```

# Object and Collection Initializers

- Collection initializers let you specify one or more element initializers when you initialize a collection class that implements `IEnumerable` or a class with an `Add` extension method. The element initializers can be a simple value, an expression or an object initializer. By using a collection initializer you do not have to specify multiple calls to the `Add` method of the class in your source code; the compiler adds the calls.

```
class Cat
{
    // Auto-implemented properties.
    public int Age { get; set; }
    public string Name { get; set; }
}

static void Main()
{
    Cat cat = new Cat { Age = 10, Name = "Fluffy" };

    List<Cat> cats = new List<Cat>
    {
        new Cat(){ Name = "Sylvester", Age=8 },
        new Cat(){ Name = "Whiskers", Age=2 },
        new Cat(){ Name = "Sasha", Age=14 }
    };

    List<Cat> moreCats = new List<Cat>
    {
        new Cat(){ Name = "Furrytail", Age=5 },
        new Cat(){ Name = "Peaches", Age=4 },
        null
    };

    // Display results.
    System.Console.WriteLine(cat.Name);

    foreach (Cat c in cats)
        System.Console.WriteLine(c.Name);

    foreach (Cat c in moreCats)
        if (c != null)
            System.Console.WriteLine(c.Name);
        else
            System.Console.WriteLine("List element has null value.");
}

// Output:
//Fluffy
//Sylvester
//Whiskers
//Sasha
//Furrytail
//Peaches
//List element has null value.
```

# Inheritance

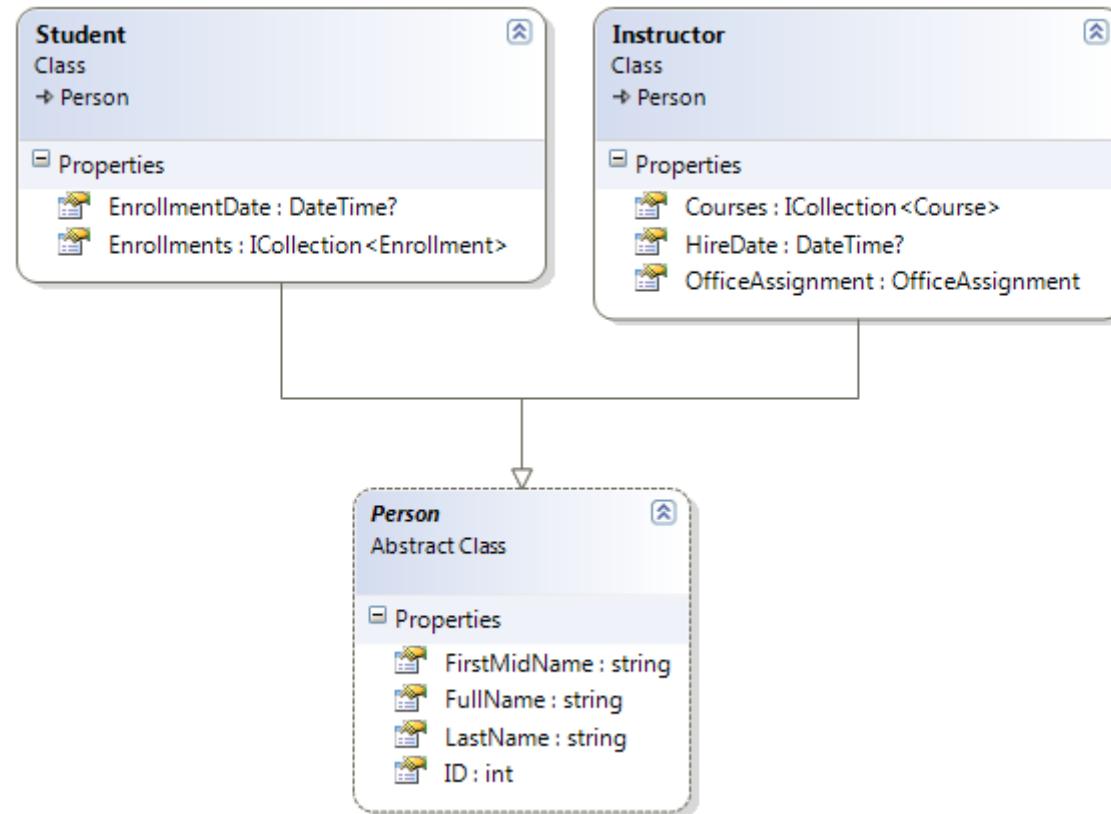
- Inheritance enables you to create a new class that reuses, extends, and modifies the behavior that is defined in another class. The class whose members are inherited is called the *base class*, and the class that inherits those members is called the *derived class*. However, all classes in C# implicitly inherit from the [Object](#) class that supports .NET class hierarchy and provides low-level services to all classes.

To inherit from a base class:

C#

```
class DerivedClass:BaseClass{}
```

# Inheritance Example



# Exercices-Class

- Define a Domino class that can instantiate objects simulating parts of a Game of dominoes. The constructor of this class will initialize the values of the points present on The two faces A and B of the domino (default values = 0). Two other methods will be defined:
  - a display\_points () method that displays the points on both sides;
  - a value () method that returns the sum of the points on both sides.

# Exercices- Inheritance

- Set a Circle class. The objects constructed from this class will be circles of various sizes. In addition to the constructor method, you need an area method, which must return the area of the circle.
- Then define a Cylinder class derived from the previous one. The constructor of this new class has two parameters: radius and height. You add a volume method that returns the cylinder volume (reminder: volume of a cylinder = area of section  $\times$  height).
- Complete the previous exercise by adding another class Cone, which must derive this time from the class Cylinder, and not the constructor will also include the two parameters radius and height. This new class will have its own volume method, which must return the volume of the cone (recall: volume of a cone = volume of the corresponding cylinder divided by 3).

# Challenges

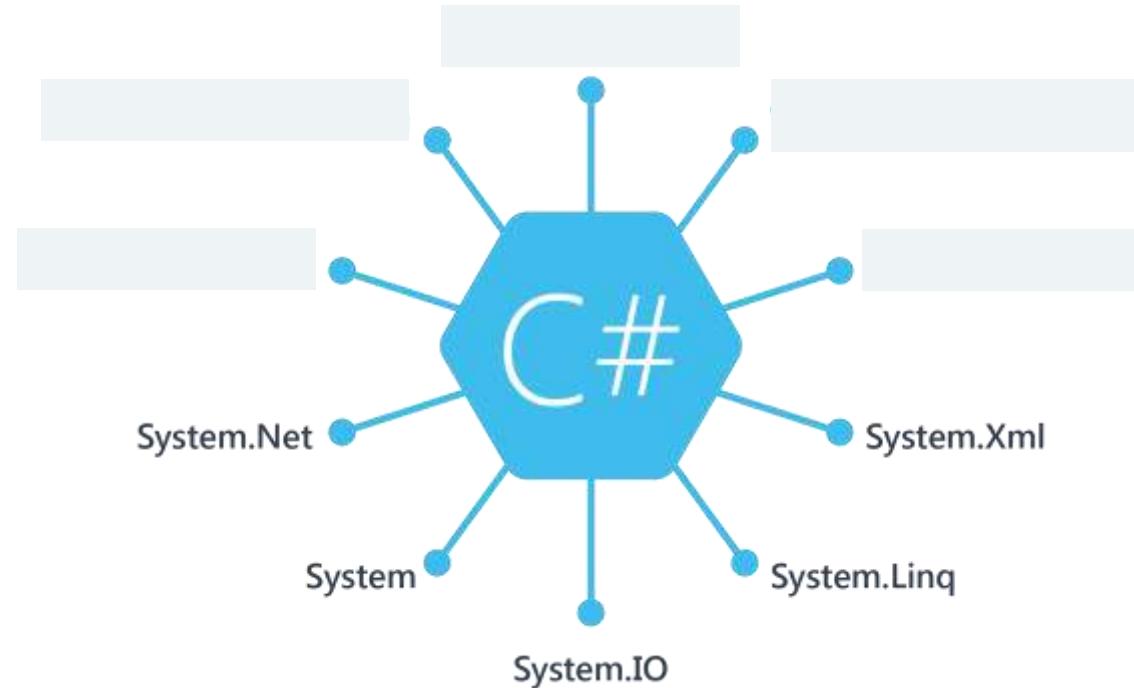
- Work on developping a C# console application by at max a group of two students. The Topic muts be choosen between the eigth proposed.

# Webography

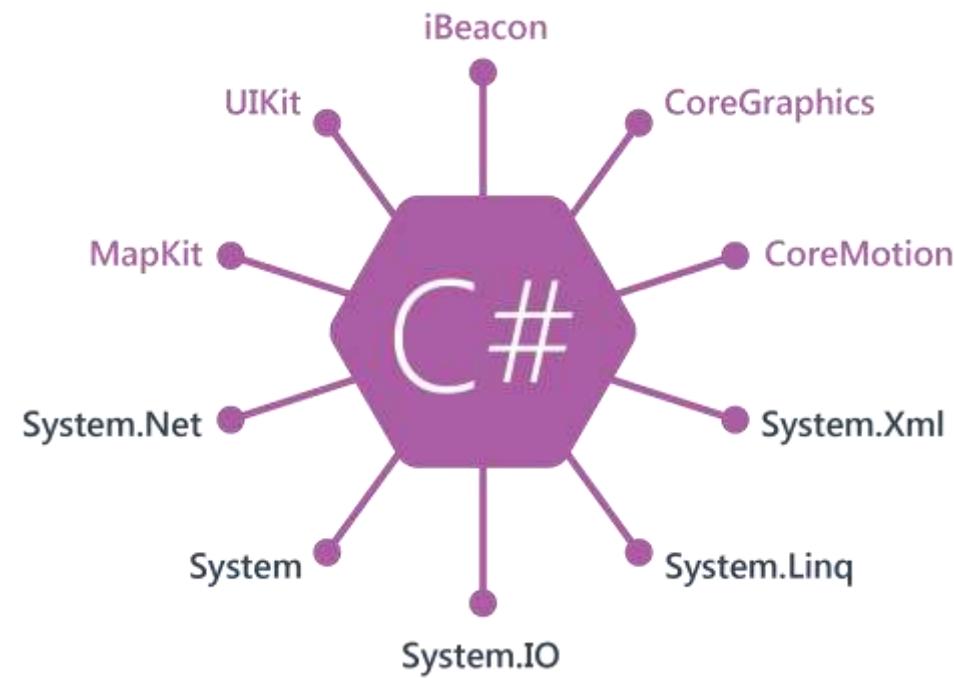
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://benjii.me/>
- [http://www.w3schools.com/asp/webpages\\_razor.asp](http://www.w3schools.com/asp/webpages_razor.asp)
- <https://blogs.msdn.microsoft.com/webdev/>
- [https://www.tutorialspoint.com/asp.net\\_core/](https://www.tutorialspoint.com/asp.net_core/)
- [https://www.youtube.com/results?search\\_query=ASP.NET+Core+2017](https://www.youtube.com/results?search_query=ASP.NET+Core+2017)

# Comment Xamarin fonctionne

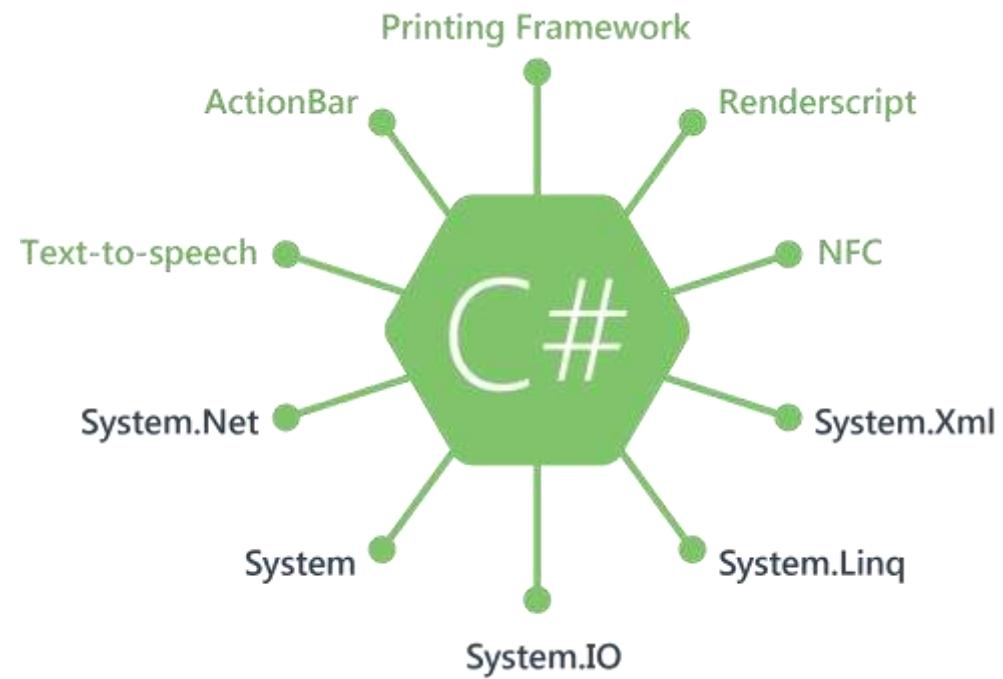
# .NET + Windows APIs



# .NET + iOS APIs | 100% Coverage



# .NET Android APIs | 100% Coverage

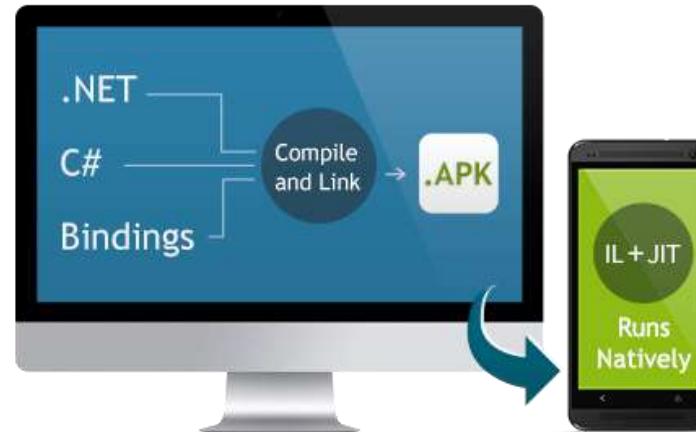


Tout ce que vous pouvez faire en Objective-C, Swift ou Java peut être fait en C# avec Xamarin et Visual Studio.

# Performance native



Xamarin.iOS compile votre application en binaire ARM pour l'App Store d'Apple.



Xamarin.Android utilise un système Just In Time lors de l'exécution.

# Toujours à jour



Support le jour de la sortie :  
iOS 5, iOS 6, iOS 7, iOS 7.1, iOS 8



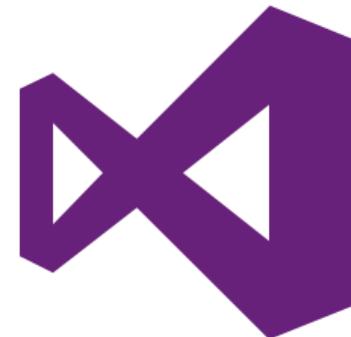
Support complet :

- Google Glass
- Android Wear
- Amazon Fire TV
- Et plus !

# Environnement de développement

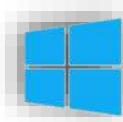
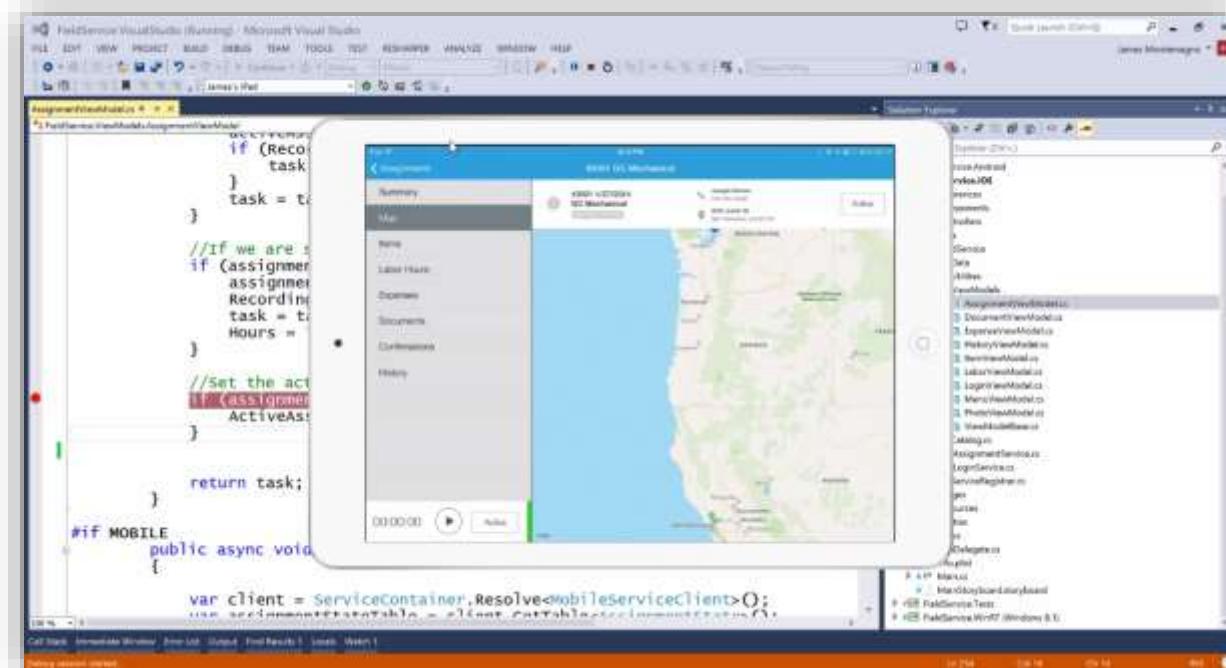


Xamarin Studio  
PC ou Mac



Visual Studio Plugin  
VS 2010 et Plus

# Intégration dans Visual Studio



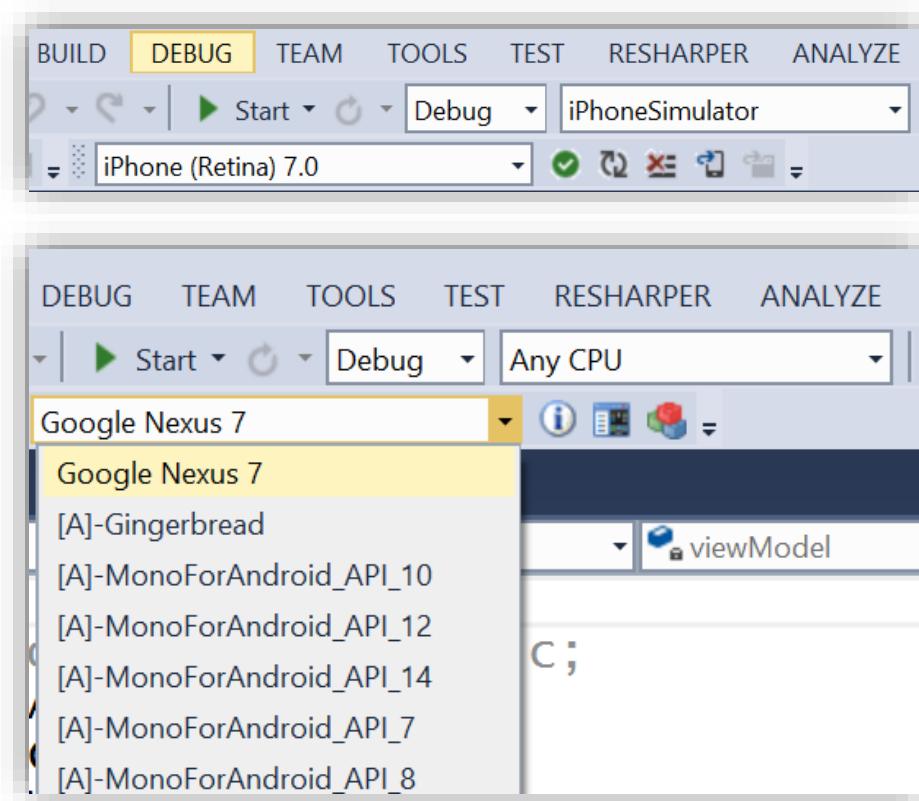
**Une seule solution :**

- iOS
- Android
- Windows Phone
- Windows Store

**Utilisez  
l'écosystème  
Microsoft :**

- ReSharper
- Team Foundation Server
- Tous vos outils préférés

# Intégration dans Visual Studio

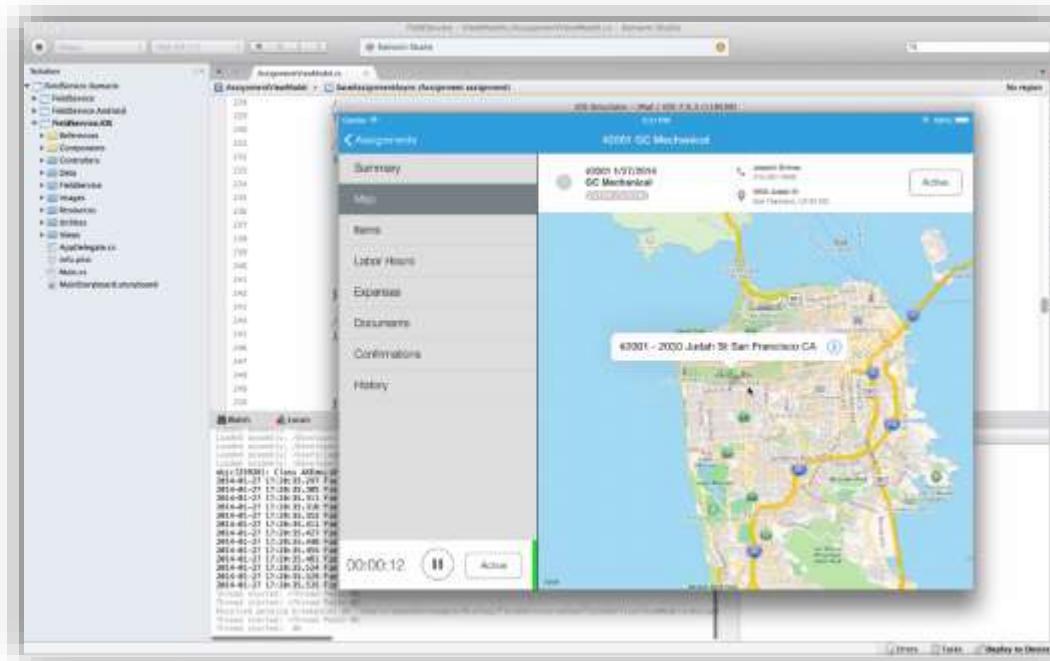


Déboguez sur :

- Émulateurs
- Devices

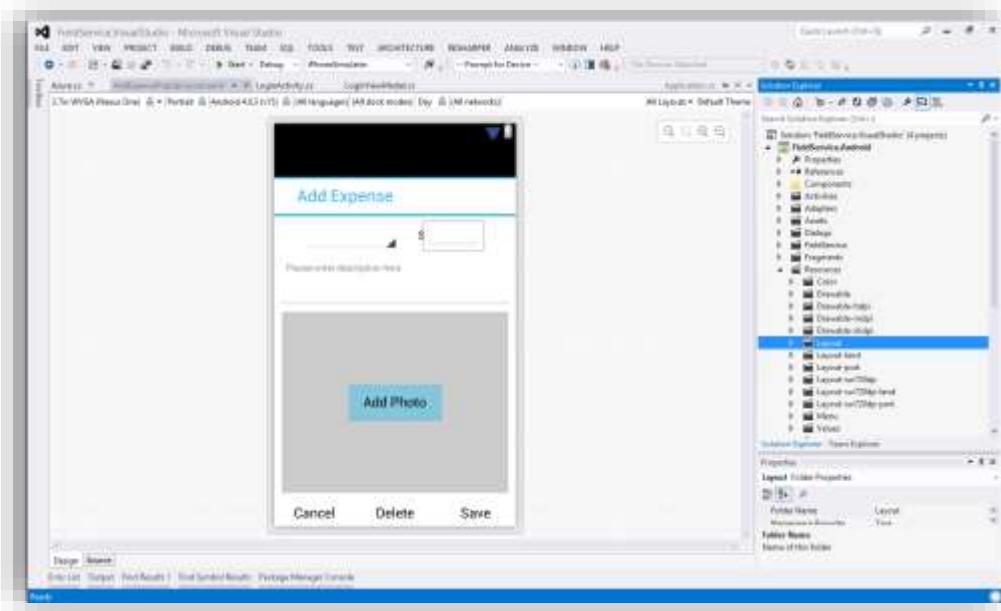
# Xamarin Studio

- Optimisé pour le développement mobile cross-platform
- Explorez les APIs avec l'auto compléction
- Designers pour Android et iOS
- Débogage puissant sur émulateur ou device

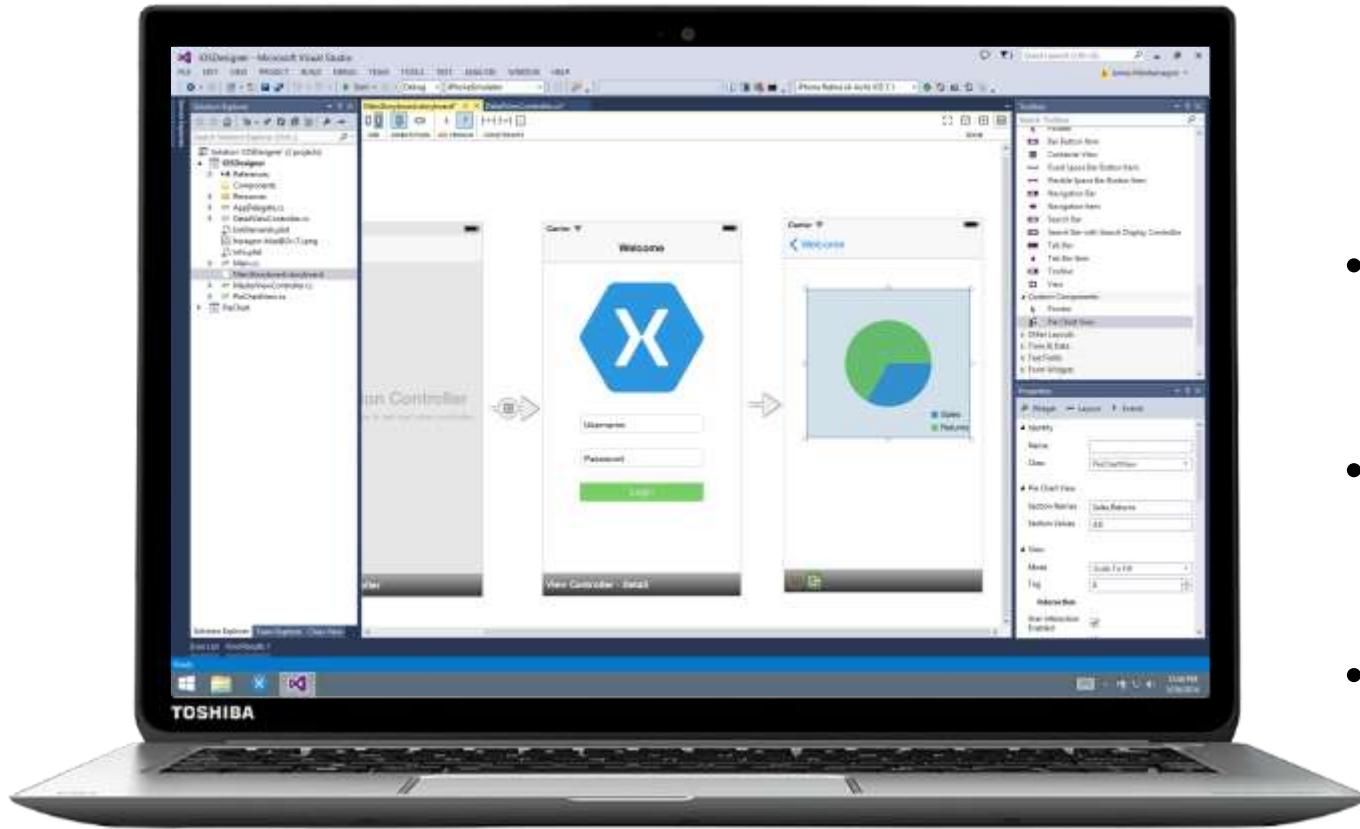


# Designer Android

- Disponible dans :
  - Xamarin Studio
  - Visual Studio
- Créez votre UI avec un simple drag & drop
- Ciblez des écrans, des résolutions et des versions d'Android différentes
- Les layouts sont au format XML standard d'Android

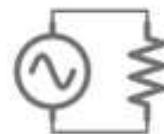
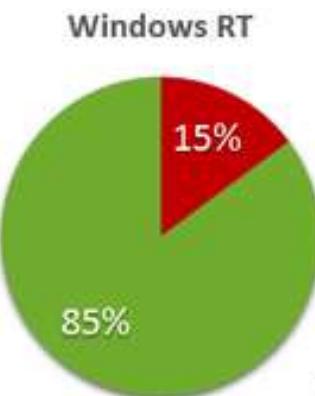
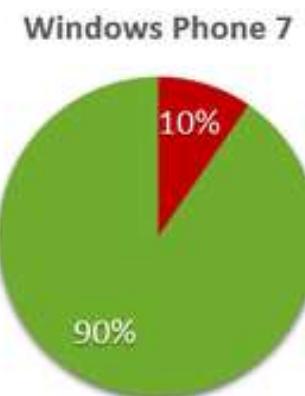
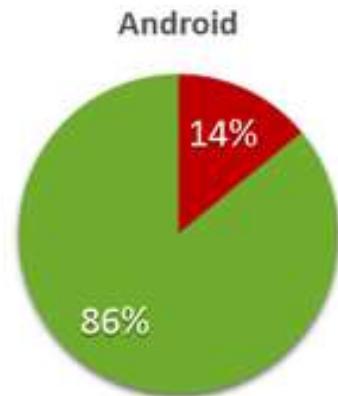
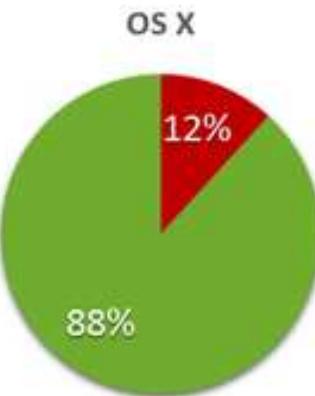
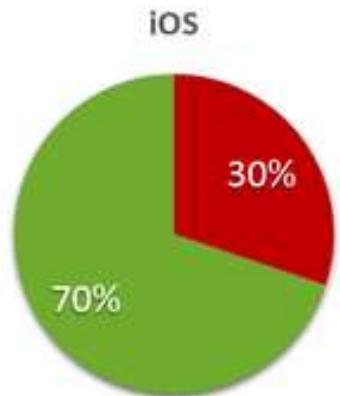


# Xamarin Designer pour iOS



- Disponible dans Xamarin Studio et Visual Studio
- Support de tous les éléments UIKit
- Support de composants tiers
- Aperçu des changements en temps réel

# Partage de code : Accélérer le développement



iCircuit Code Reuse  
February 5, 2013  
Copyright 2013 Krueger Systems, Inc.

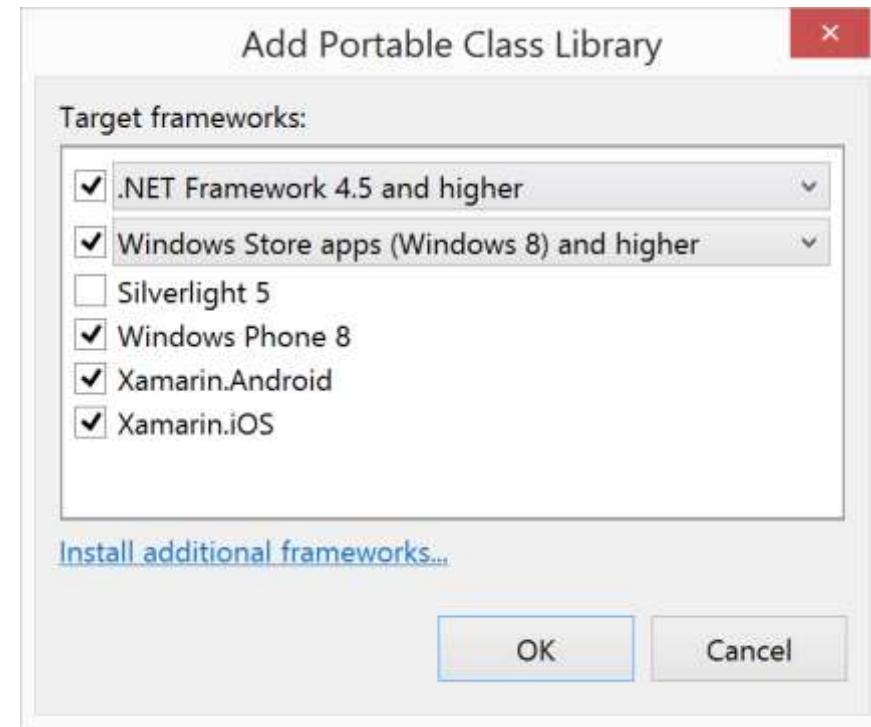
Source: <http://praeclarum.org/post/42378027611/icircuit-code-reuse-part-cinq>

# Portable Class Libraries

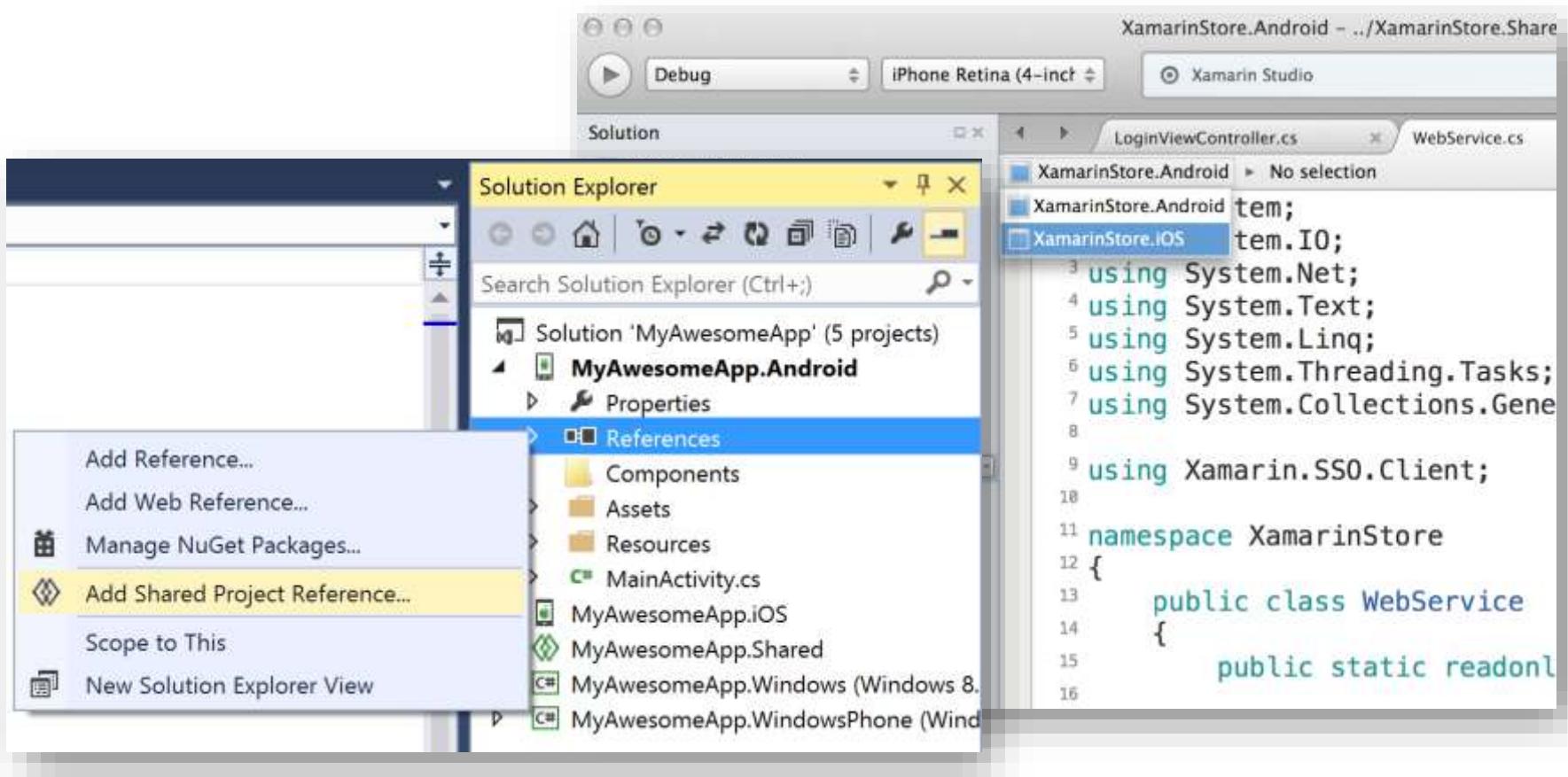
Les bibliothèques de classe portables permettent d'écrire votre code C # et le partager sur toutes les plates-formes

Ajoutez maintenant un support officiel pour créer et utiliser des PCL dans Visual Studio et Xamarin Studio

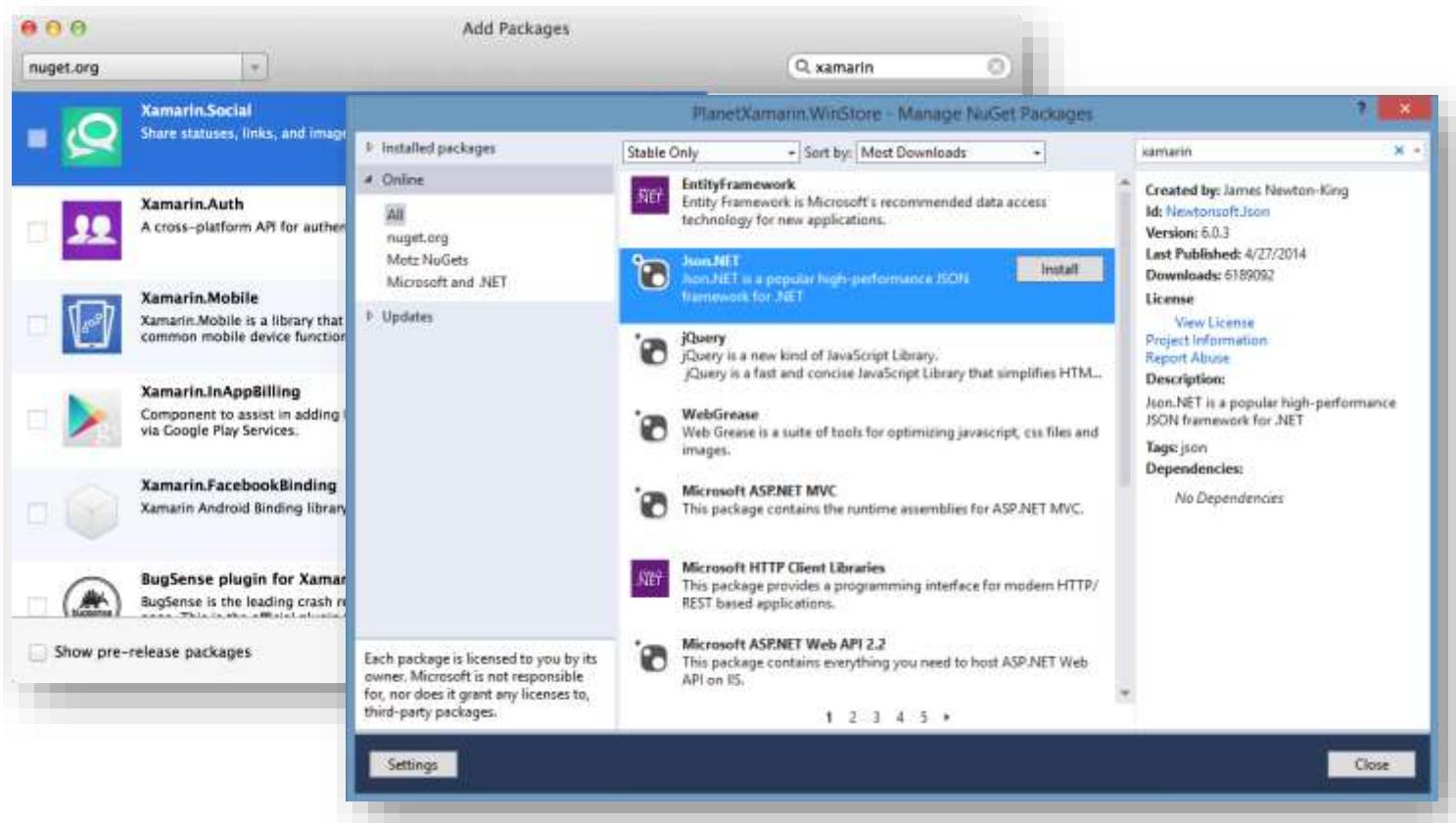
- 1 Assembly
- Plusieurs plateformes
- Incluant :
  - **Xamarin.Android**
  - **Xamarin.iOS**



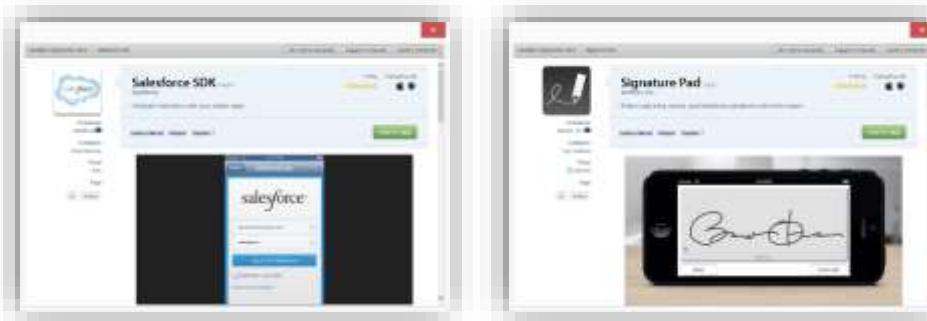
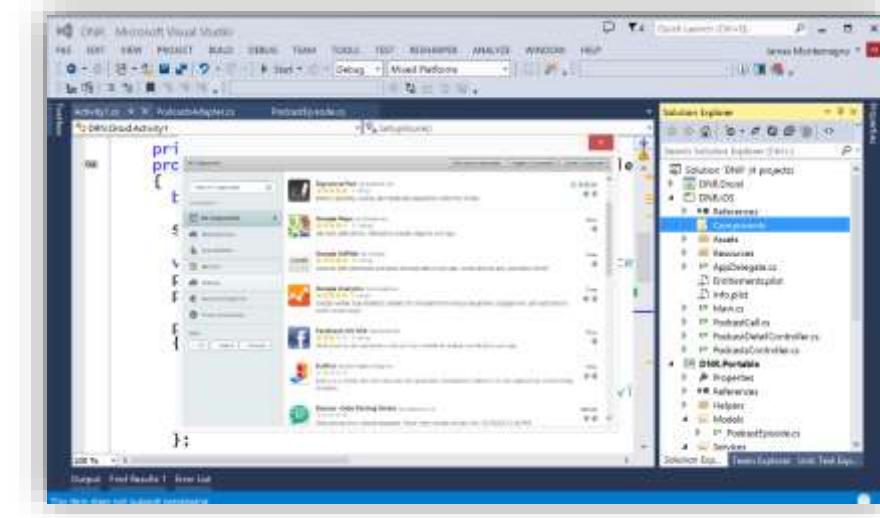
# Shared Projects



# NuGet



# Xamarin Component Store

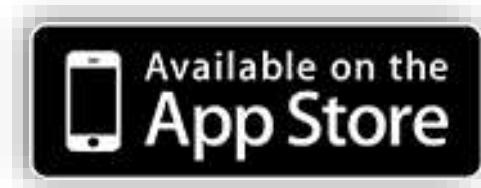


Créez plus vite

- Ajoutez des composants existants depuis Visual Studio ou Xamarin Studio
- Des éléments d'UI, d'intégration avec un service cloud et d'autres sont disponibles

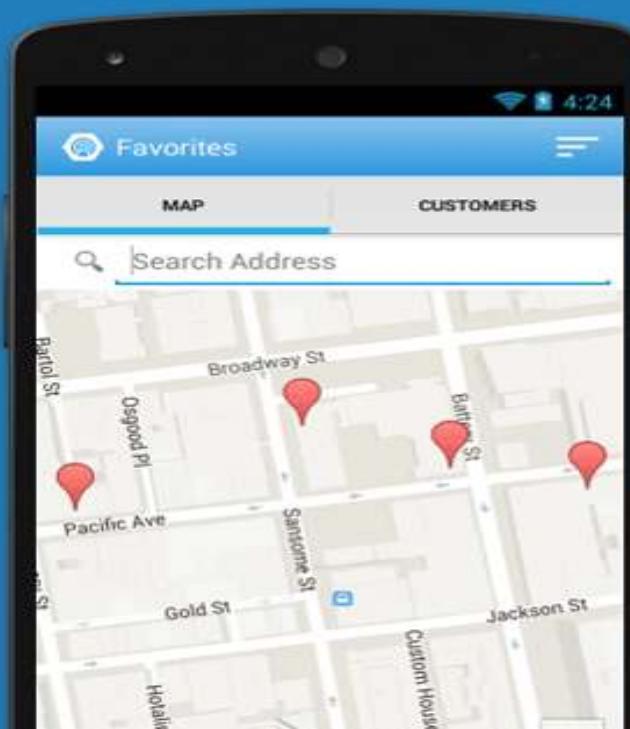
# Distribuable partout

Une application Xamarin peut être distribuée partout

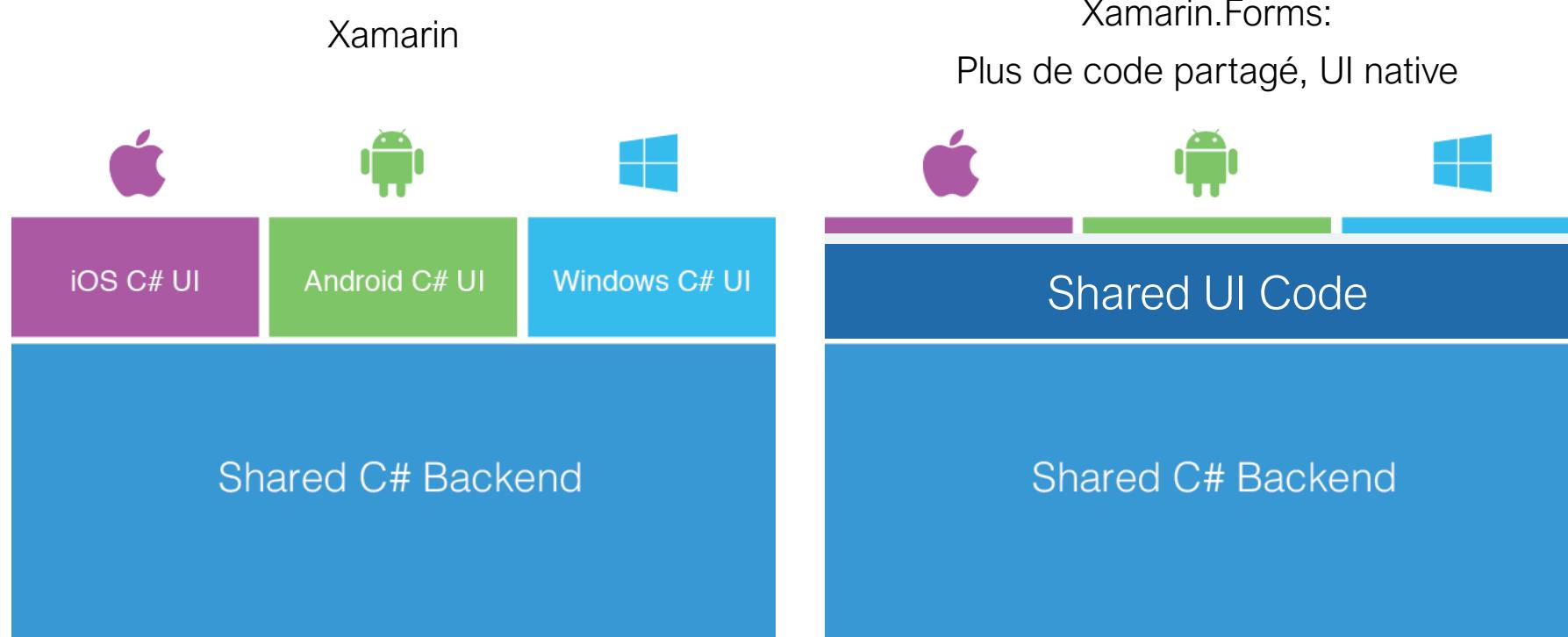


# Meet Xamarin.Forms

Build native UIs for iOS, Android and Windows Phone from a single, shared C# codebase.



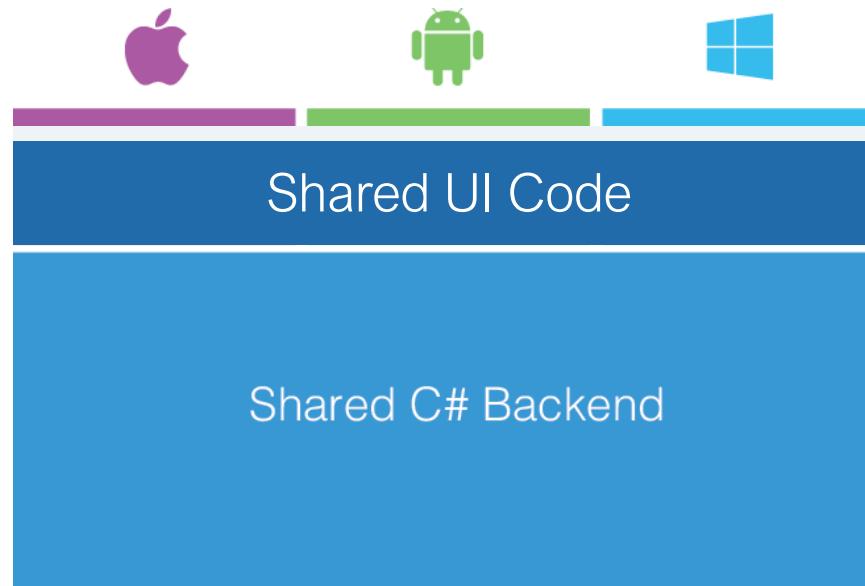
# L'approche de Xamarin.Forms



Xamarin.Forms est une boîte à outils de classes d'interface utilisateur multiplateformes construites au-dessus des classes d'interface utilisateur spécifiques à la plateforme plus fondamentales: Xamarin.Android et Xamarin.iOS. Xamarin.Android et Xamarin.iOS fournissent des classes mappés à leurs SDKs UI natives respectives: iOS UIKit et SDK Android. Xamarin.Forms se lie également directement à la SDK native Windows Phone. Ceci fournit un ensemble de composants d'interface utilisateur qui rendent dans chacun des trois systèmes d'exploitation natifs.

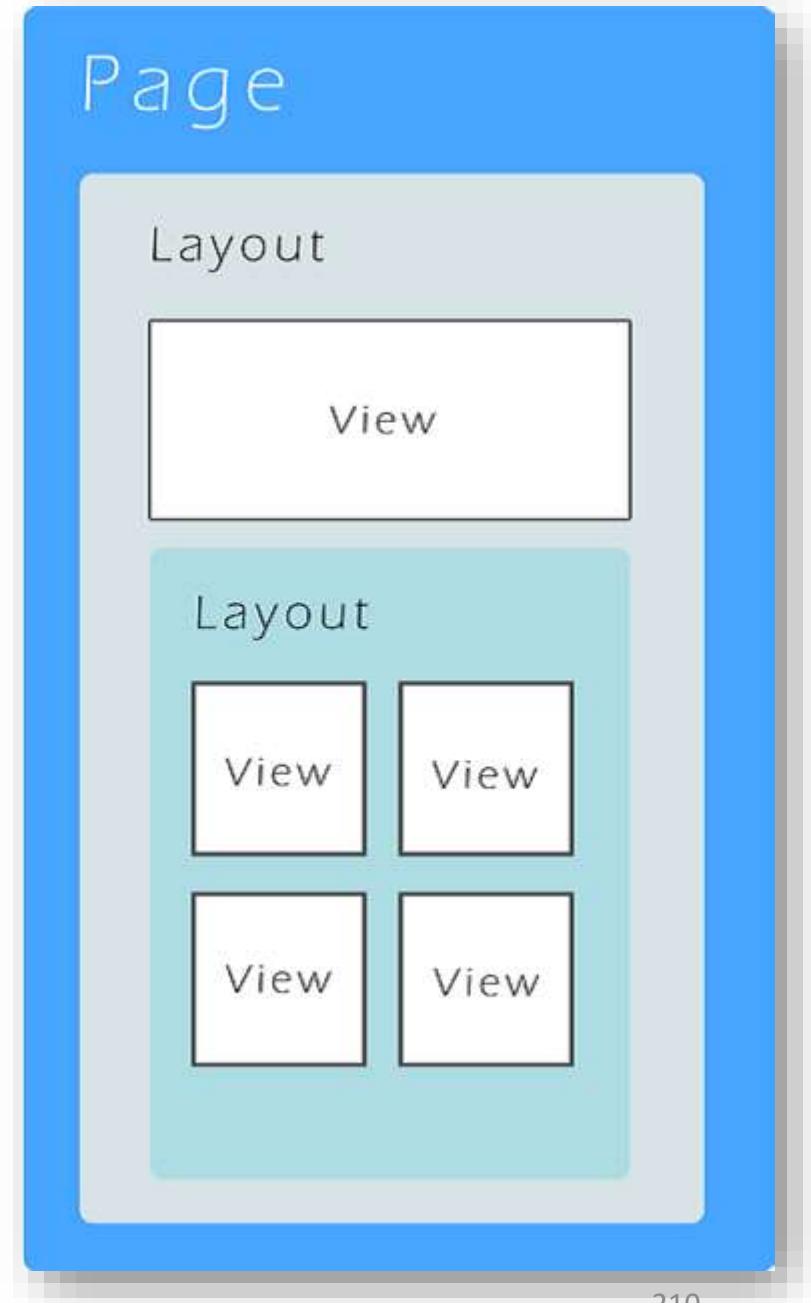
# De quoi dispose-t-on ?

- +40 Pages, Layouts, et Controls utilisables en C# ou Xaml
- Two-way Data Binding
- Navigation
- Animation API
- Dependency Service
- Messaging Center



# Xamarin.forms pour l'interface utilisateur

- Les **Pages**, **Layouts**, et les **Views** constituent le noyau de l'interface utilisateur **Xamarin.Forms** (Voir figure).
- Les pages sont le principal élément, et chaque écran est composé par une seule classe de page.
- Une **Page** peut contenir plusieurs **Layouts**, qui peuvent alors en contenir d'autres et/ou des **Views**.
- Le but des **Pages** et **Layouts** est de contenir et présenter les contrôles héritées de la classe **View**.



# Pages

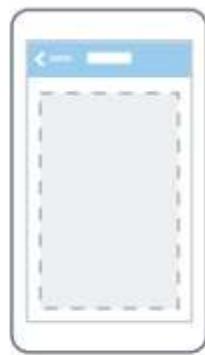
- La classe page est le conteneur primaire de chaque écran principal dans une application. Dérivée de **Xamarin.forms.VisualElement**, une page est la classe de base pour la création d'autres classes de l'interface utilisateur de haut niveau. Voici les pages primaires :



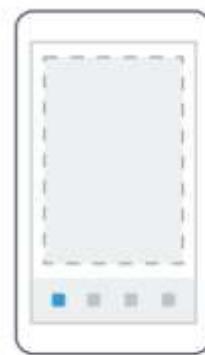
ContentPage



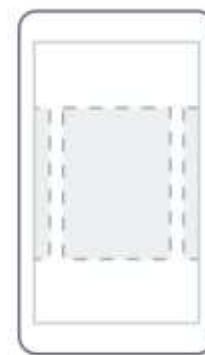
MasterDetailPage



NavigationPage



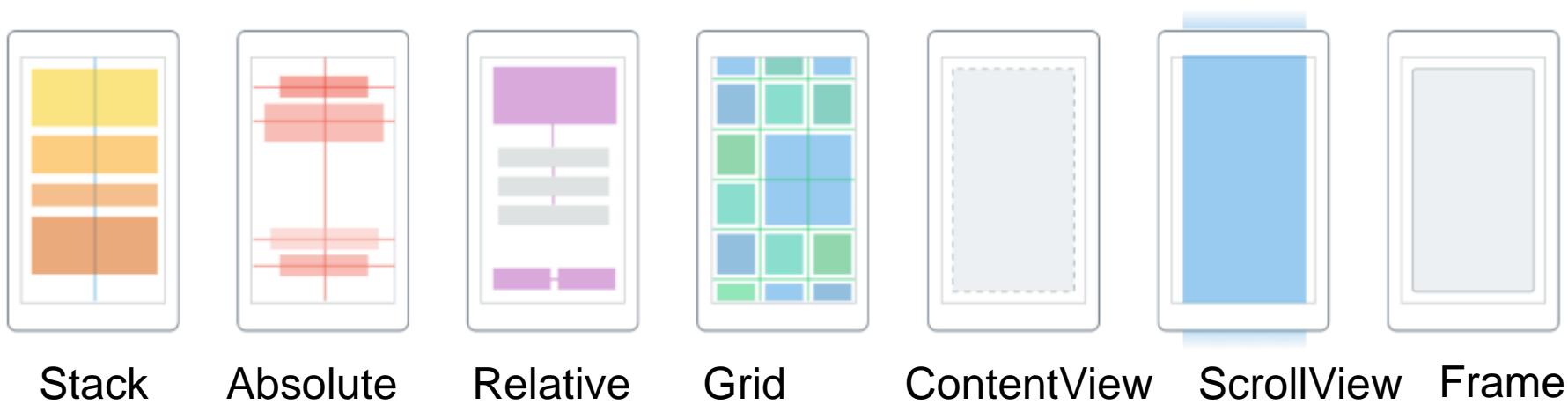
TabbedPage



CarouselPage

La classe page est le conteneur primaire de chaque écran principal dans une application. Dérivée de **Xamarin.forms.VisualElement**, une page est la classe de base pour la création d'autres classes de l'interface utilisateur de haut niveau.

# Layouts



**A l'intérieur d'une page on retrouve des Layouts :** Beaucoup d'options disponibles de quelque chose de simple comme un Stack à quelque chose de complexes et puissantes comme un Grid.

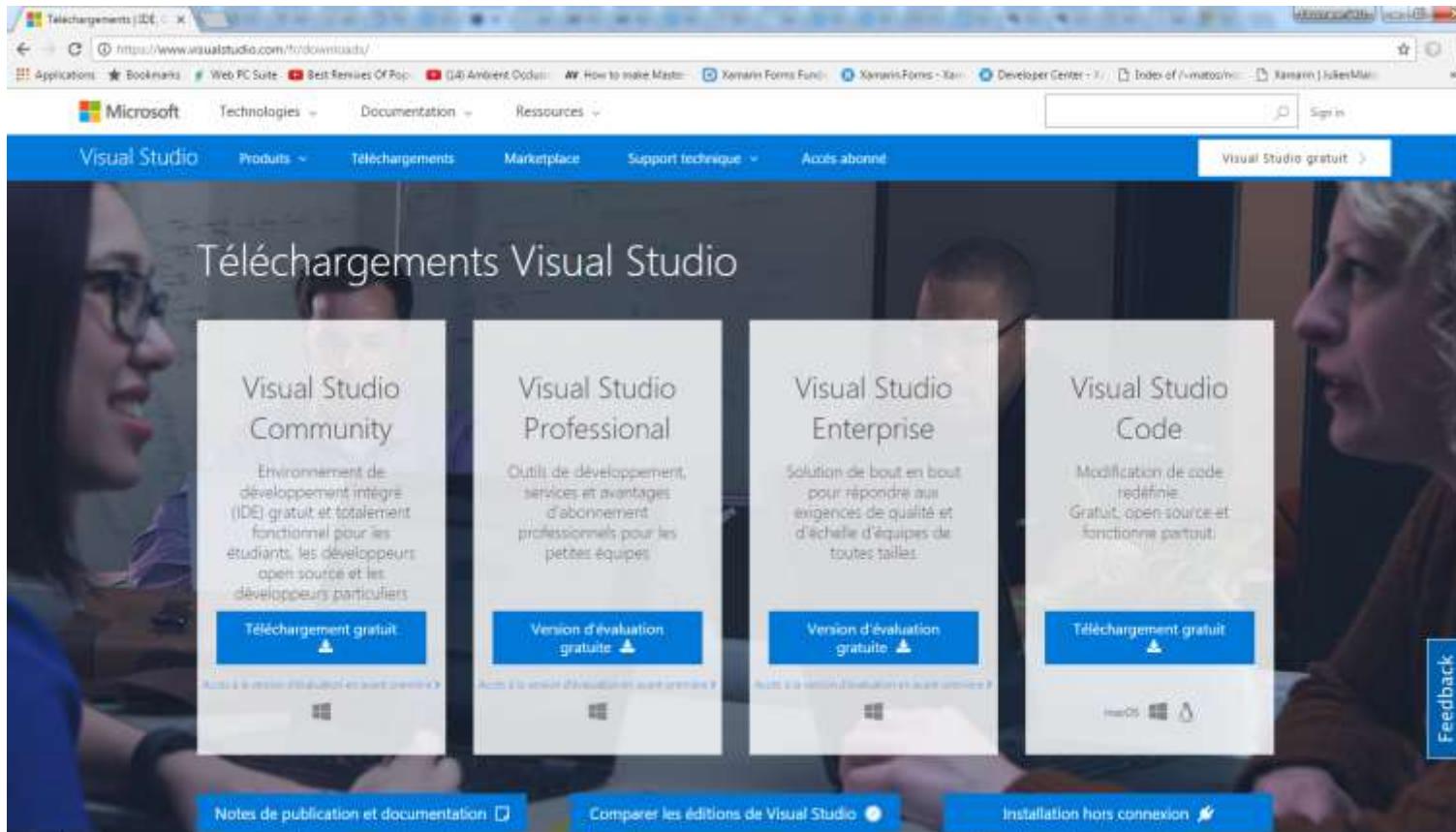
# Views

Les vues sont des contrôles, les éléments visibles et interactifs sur une page. Celles-ci vont des **vues de base** comme des **boutons, des étiquettes et des zones de texte** aux **vues plus avancées comme les listes et la navigation**. Les vues contiennent des **propriétés** qui déterminent leur **contenu, la police, la couleur et l'alignement**. L'alignement horizontal et vertical est fixé par propriétés **HorizontalOptions** et **VerticalOptions**. Comme mise en page, les vues peuvent être rembourrées avec l'espace, dimensionné pour étendre, pour remplir l'espace disponible, ou diminuer en fonction de leur contenu. Plus loin, nous allons coder des vues.

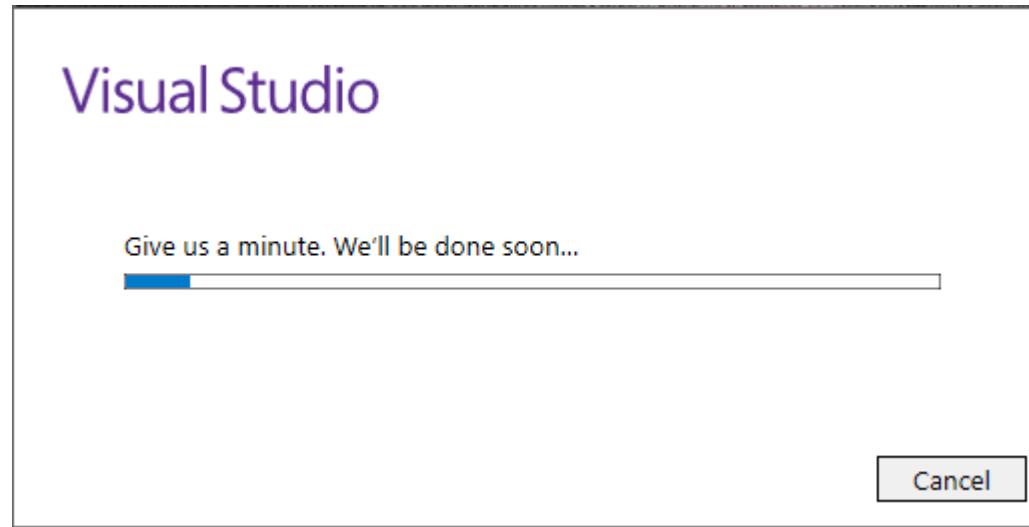


# Setup et Installation

<https://www.visualstudio.com/fr/downloads/>



# Installer Visual Studio 2017 IDE



# Installer Xamarin

Installing - Visual Studio Community 2017 (15.0.26228.4)

Workloads Individual components Language packs

Node.js development Build scalable network applications using Nodejs, an asynchronous event-driven JavaScript runtime.

Office/SharePoint development Create Office and SharePoint add-ins, SharePoint solutions, and VSTO add-ins using C#, VB, and JavaScript.

**Mobile & Gaming (5) —**

Mobile development with .NET Build cross-platform applications for iOS, Android or Windows using Xamarin.

Mobile development with JavaScript Build Android, iOS and UWP apps using Tools for Apache Cordova.

Game development with C++ Build cross-platform applications for iOS, Android or Windows using C++.

Game development with C# Create 2D and 3D games with Unity, a powerful cross-platform development environment.

Location  
C:\Program Files (x86)\Microsoft Visual Studio\2017\Community ...

Summary

Mobile development with .NET

Included

- ✓ Xamarin
- ✓ .NET Framework 4.6.1 development ...
- ✓ C# and Visual Basic
- ✓ .NET Portable Library targeting pack

Optional

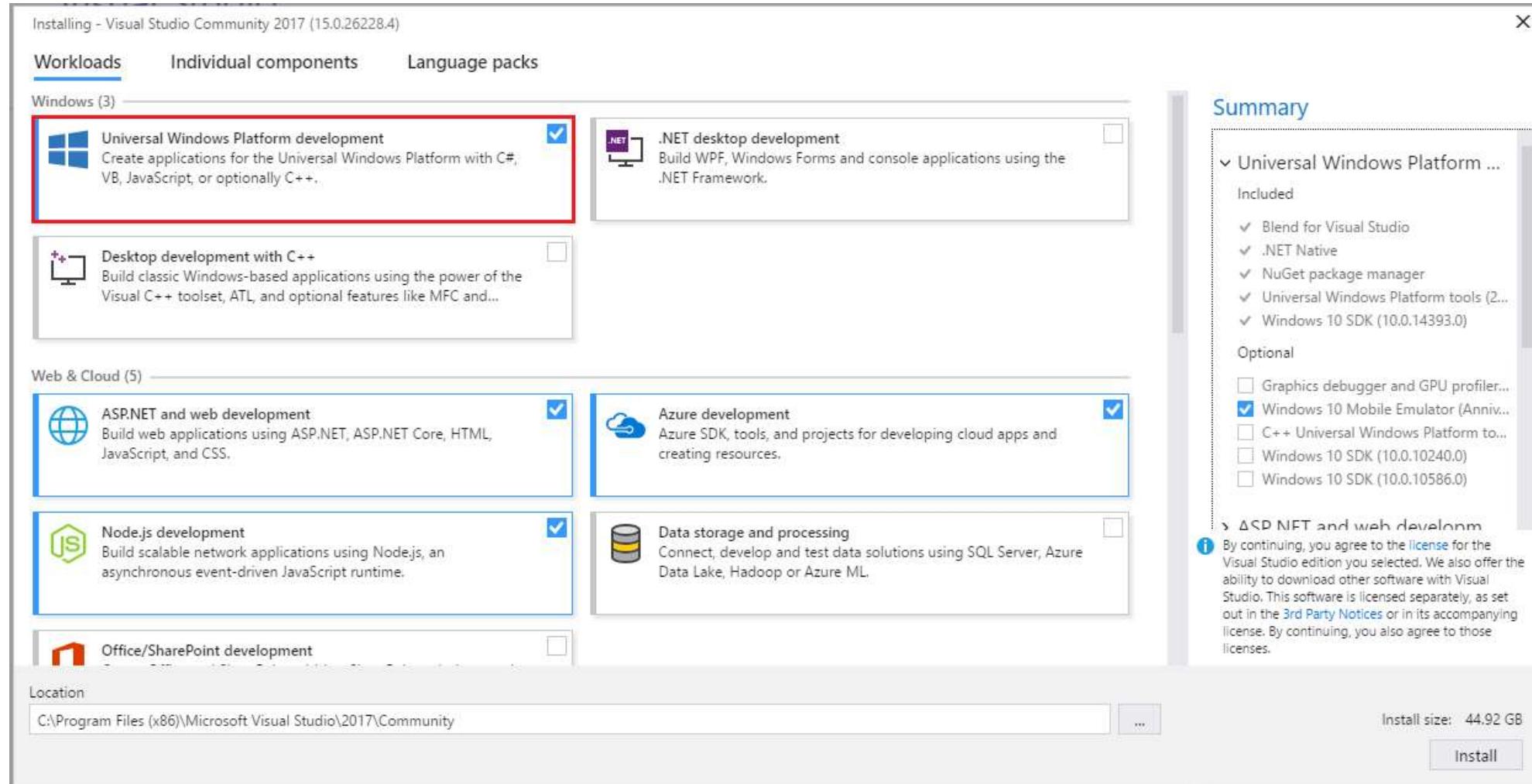
- ✓ Xamarin Workbooks
- ✓ Android NDK (R13B)
- ✓ Android SDK setup (API level 23)
- ✓ Java SE Development Kit (8.0.920.14)
- ✓ Google Android Emulator (API Level...)
- ✓ F# language support
- ✓ Intel Hardware Accelerated Executio...
- ✓ Windows 10 Mobile Emulator (Anniv...
- ✓ Universal Windows Platform tools fo...

By continuing, you agree to the [license](#) for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the [3rd Party Notices](#) or in its accompanying license. By continuing, you also agree to those licenses.

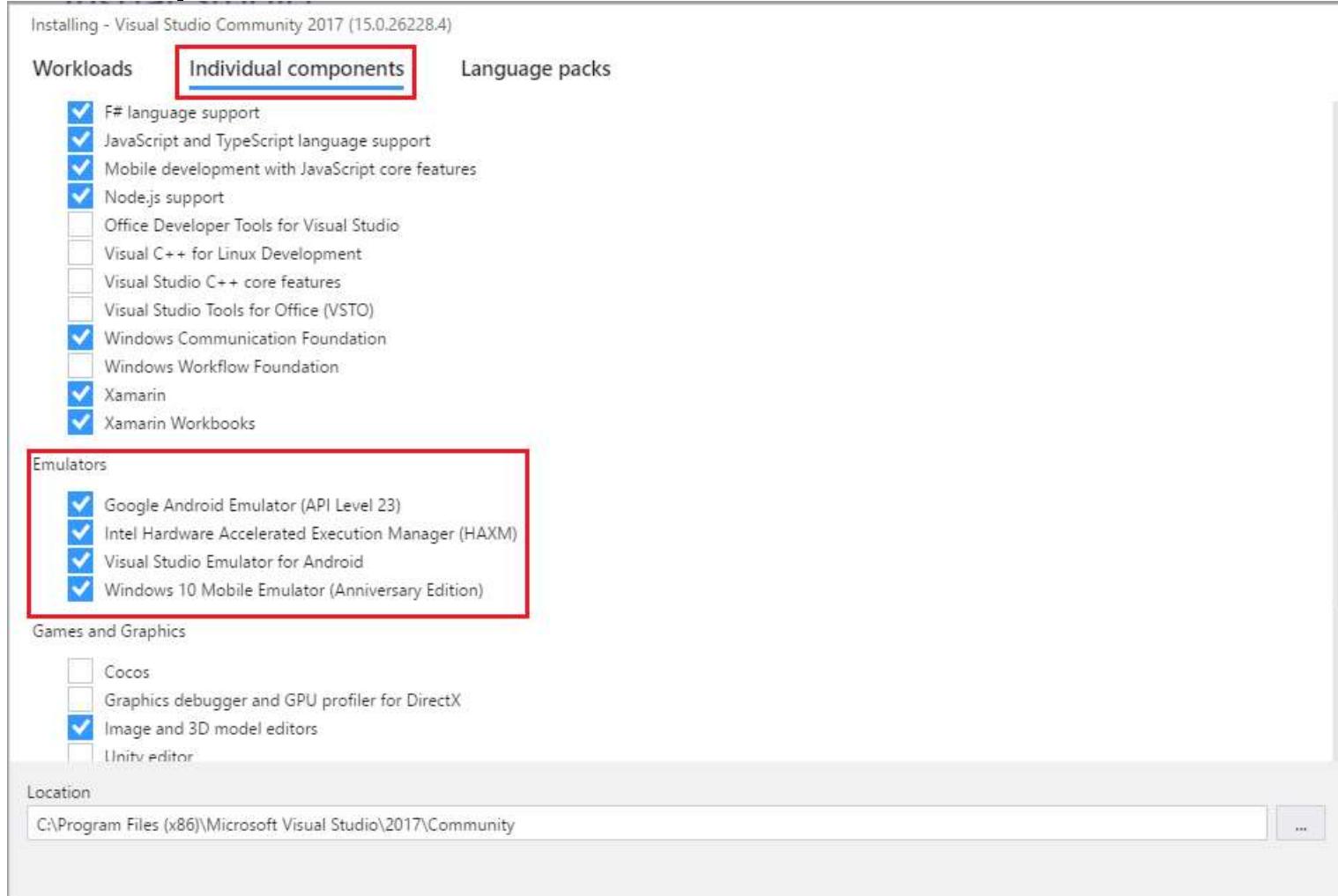
Install size: 44.46 GB

Install

# Installer Visual Studio 2017 pour Windows Universal Platform Development



# Installer les Emulateurs de Visual Studio Emulators pour Android et Windows 10



# Installer Visual Studio 2017 pour Windows

## Visual Studio

Products

Installed



Visual Studio Community 2017

Acquiring Microsoft.VisualStudio.Xamarin.Forms

46%

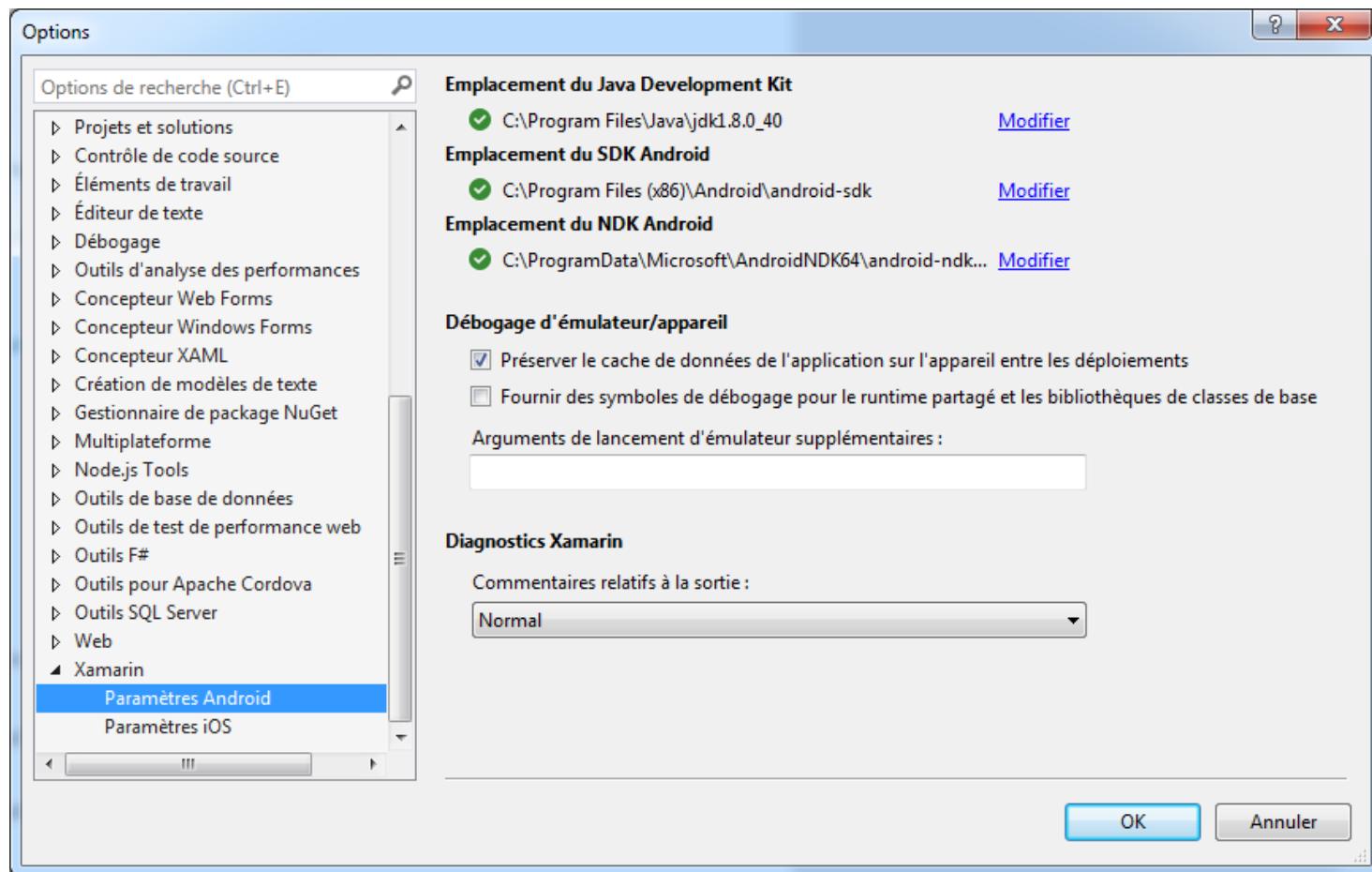
Applying Win10SDK\_10.0.14393.795

10%

Cancel

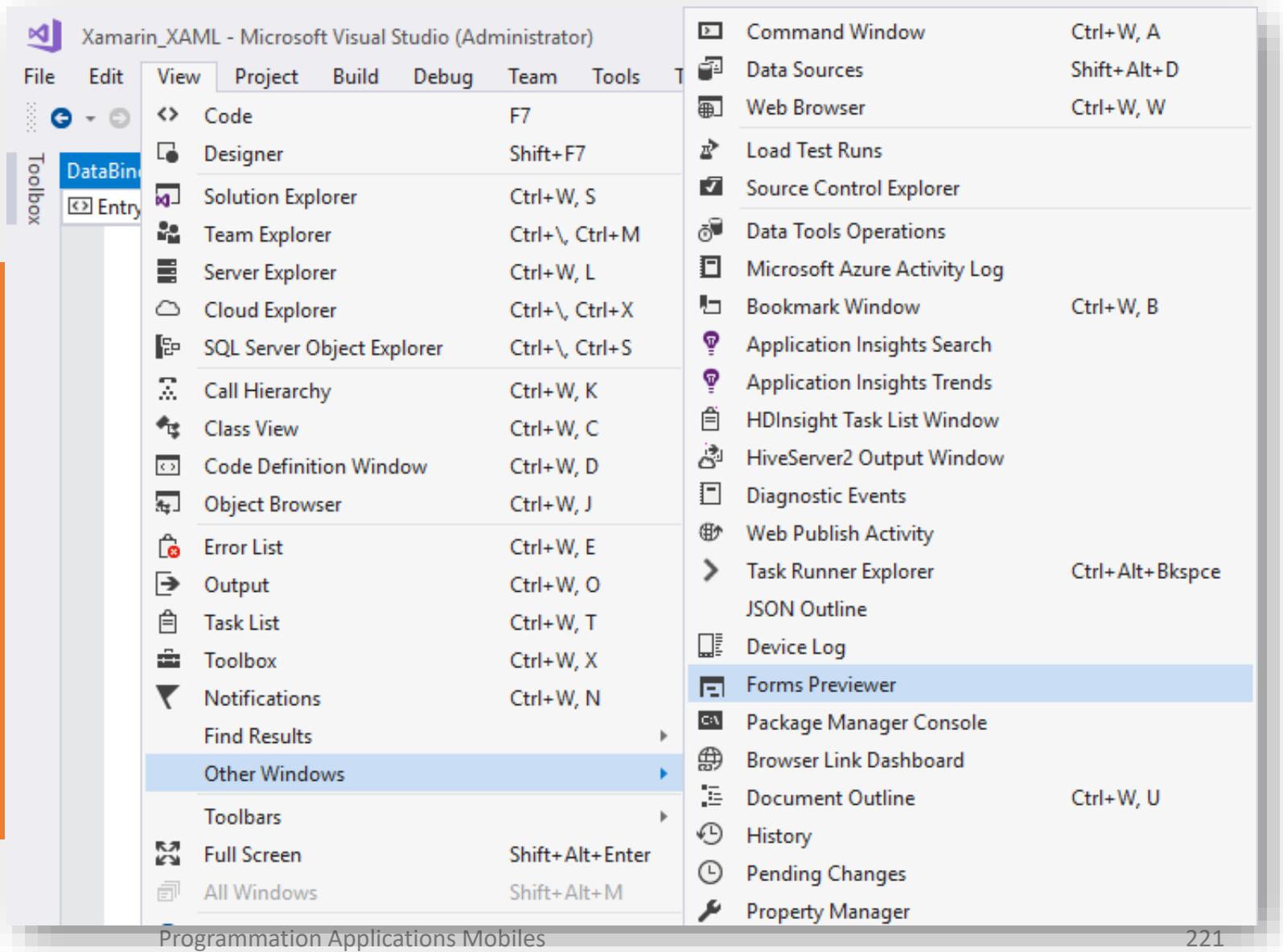
# Configurer et Installer

Pour configurer Visual Studio tools, aller à **Outils > Options > Xamarin > Paramètres Android**:



# Visual Studio 2017 Xamarin.Forms Preview supports

Avec Visual Studio 2017 et la dernière version de Visual Studio pour Mac, Xamarin.Forms XAML page, vous pouvez prévisualiser dans Xamarin.Forms pour voir les mises en page et la conception attendues dans Android et iOS. **Assurez-vous d'avoir installé le dernier Java 1.8 x64 pour l'aperçu Android.**



# Exercice 2

- Quelles sont les approches pour le développement mobile ?
- Quelles sont les motivations liées au développement Cross-Platform ?
- Qu'est-ce-que Xamarin.Forms ?
- Comment sont structurées les fenêtres créées avec Xamarin.Forms ?

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 5312 Technologies Mobiles: Développement d'Applications Mobiles Cross-Platform avec Xamarin et C#

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

**Xamarin - Une Architecture pour le Développement  
Multiplateforme**

# A propos de moi

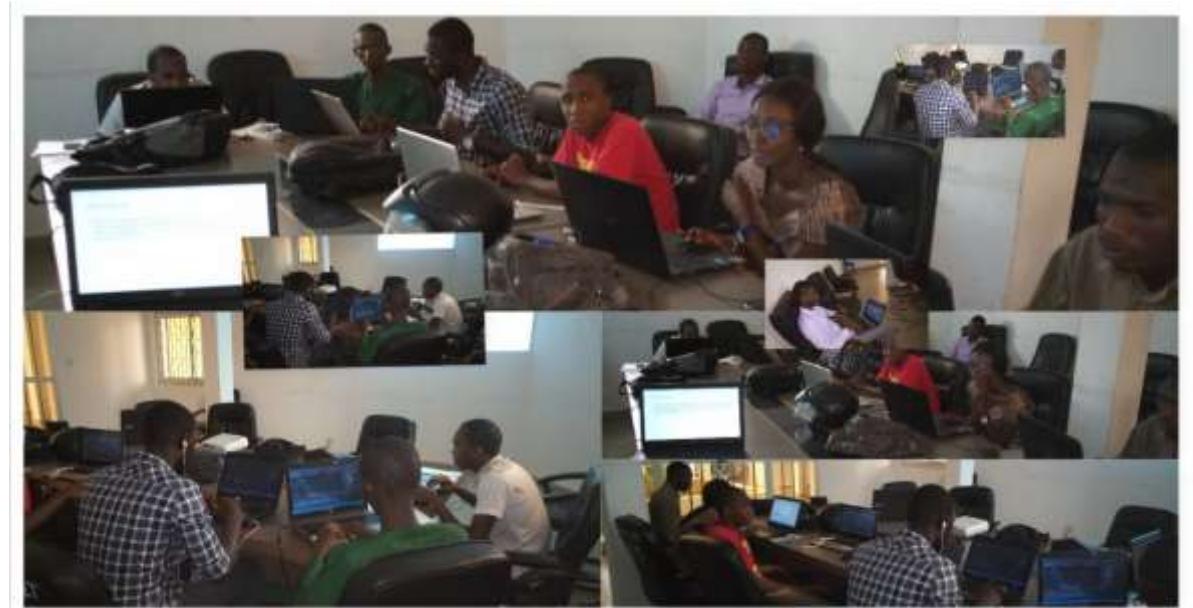


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Xamarin pour le développement d'Applications Mobiles
- 3. Une architecture pour le développement Cross-Plateforme**
4. Développement Cross-Platform avec Xamarin.Forms
5. Interface utilisateur
6. Data-Binding
7. Web Services
8. Test et Déploiement

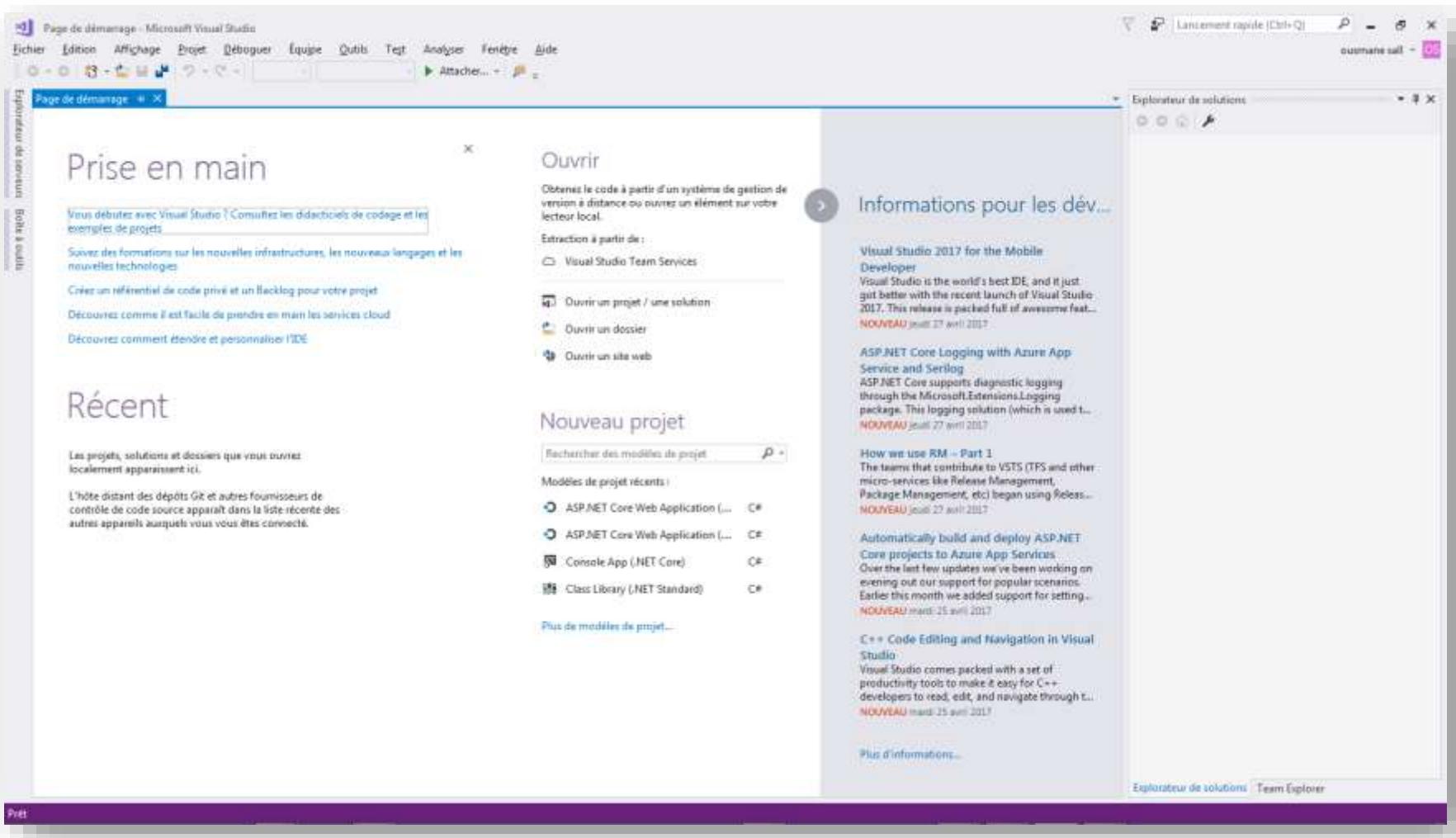
# Xamarin + Visual Studio

- Créer des applications Android, iOS et Windows Phone - de la logique métier à l'interface utilisateur - avec un code presque 100% commun utilisant Xamarin.Forms dans Visual Studio.
- Utiliser la syntaxe XAML pour construire des écrans de manière déclarative, avec des styles, des comportements et des déclencheurs.
- Cross-Platform Apps: À l'aide des «**Bibliothèques de classe portable**» ou des «**Projets partagés**», vous pouvez écrire un code commun pour partager avec les projets **Xamarin.iOS**, **Xamarin.Android** et **Xamarin.Mac**, ainsi que UWP, Windows, WPF, WinRT ou autre C # Basé sur les plates-formes à partir de Visual Studio.

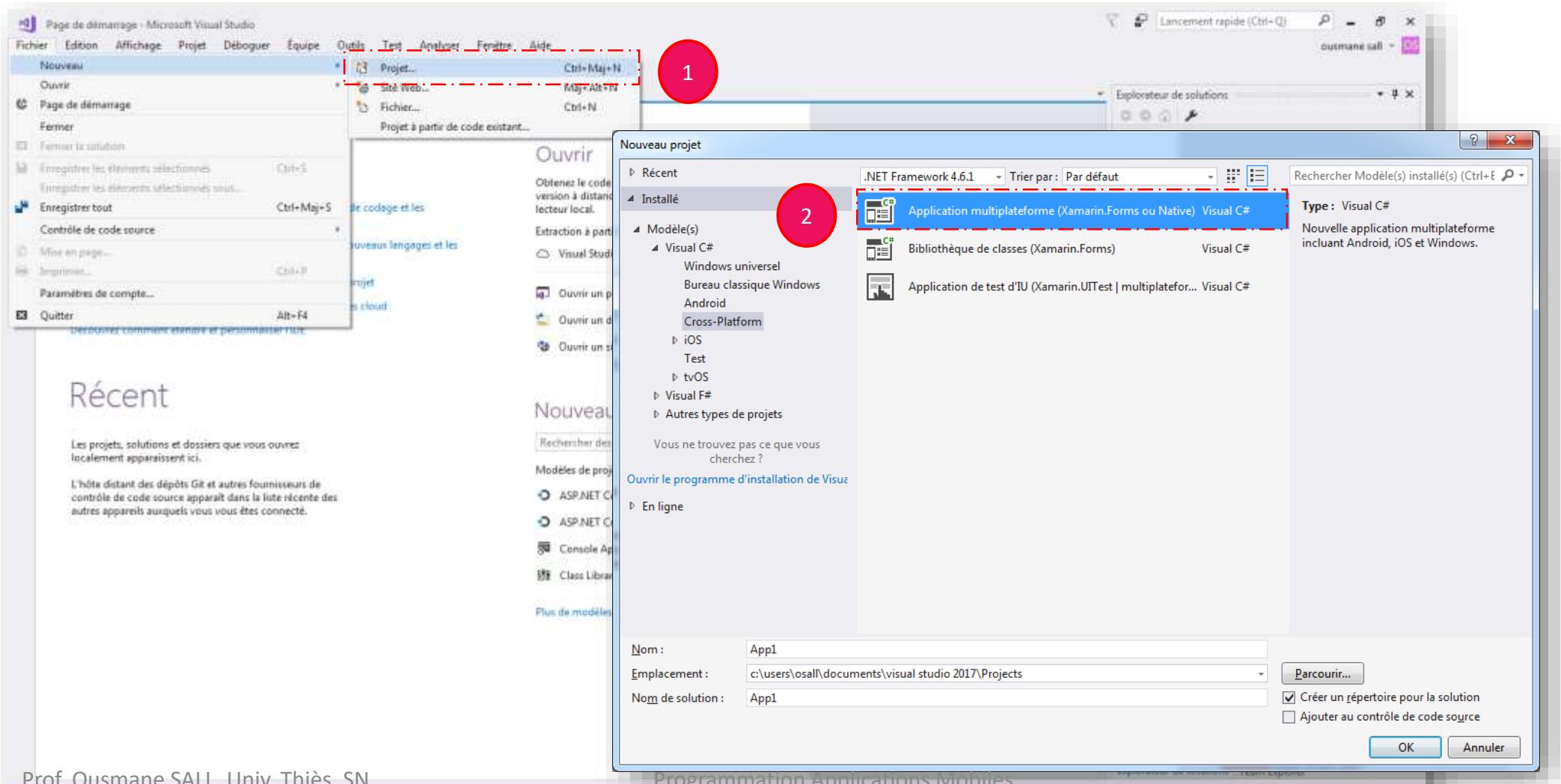
# Anatomie d'une application

- Dans Xamarin.Forms, les objets qui apparaissent sur l'écran sont collectivement appelés éléments visuels. Ils se déclinent en trois catégories principales:
  - **Page:** Une application Xamarin.Forms se compose d'une ou plusieurs pages. Une page occupe généralement tout (ou au moins une grande surface) de l'écran. Certaines applications ne comportent qu'une seule page, tandis que d'autres permettent de naviguer entre plusieurs pages.
  - **Layout:** Sur chaque page, les éléments visuels sont organisés dans une hiérarchie parent-enfant. L'enfant d'un ContentPage est généralement une disposition de sorte pour organiser les visuels. Certaine Layouts ont un enfant unique, mais de nombreuses mises en page ont plusieurs enfants que la disposition organise en elle-même. Ces enfants peuvent être d'autres Layouts ou des vues. Différents types de dispositions organisent les enfants dans une pile, dans une grille bidimensionnelle, ou de manière plus libre.
  - **View:** Le terme de vue dans Xamarin.Forms désigne différents types de présentation et les objets interactifs familiers: text, bitmaps, buttons, text-entry fields, sliders, switches, progress bars, date and time pickers, et d'autres éléments de votre propre conception. Ceux-ci sont souvent appelés contrôles ou widgets dans d'autres environnements de programmation.

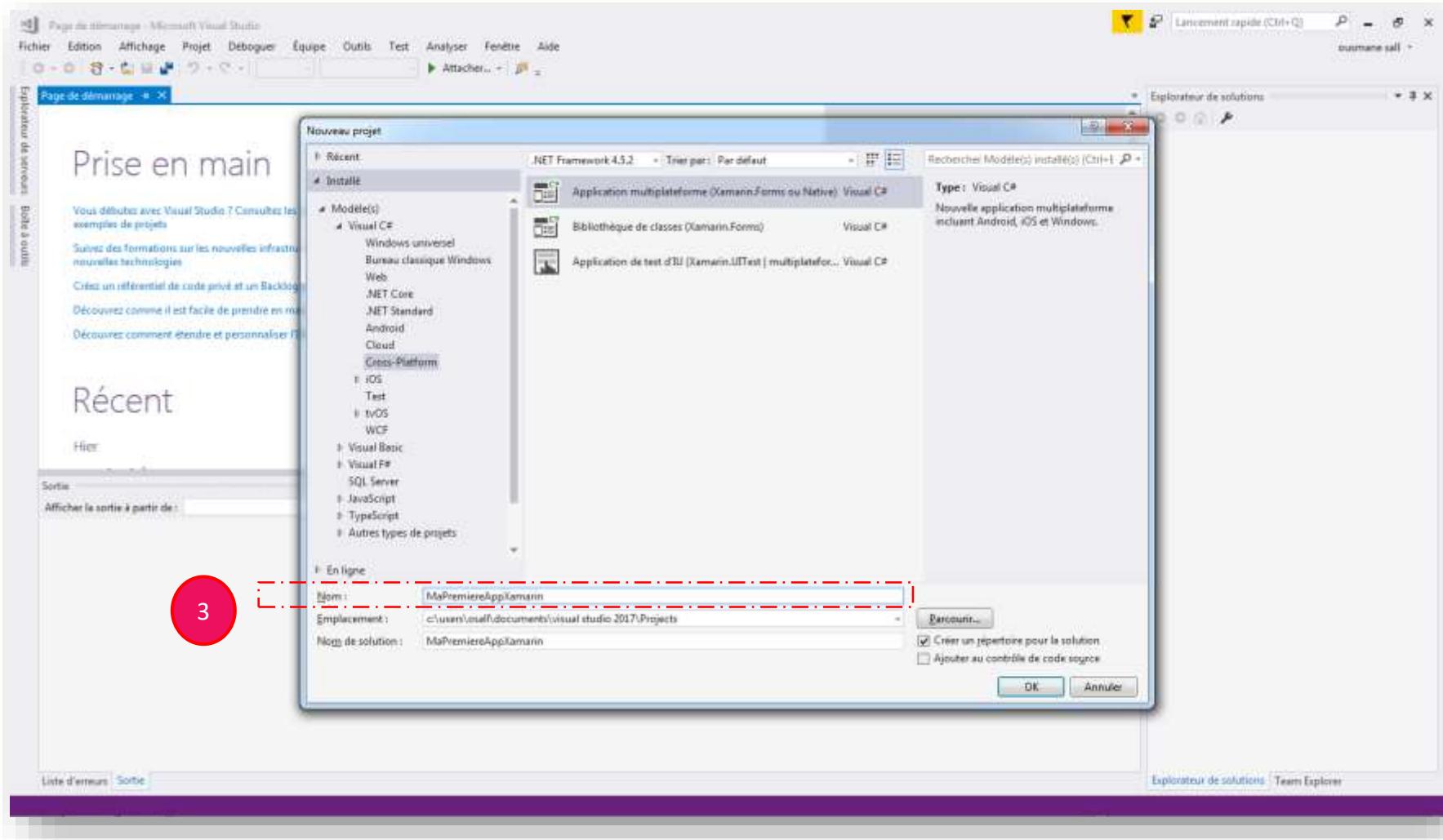
# Création d'un projet cross-plateforme



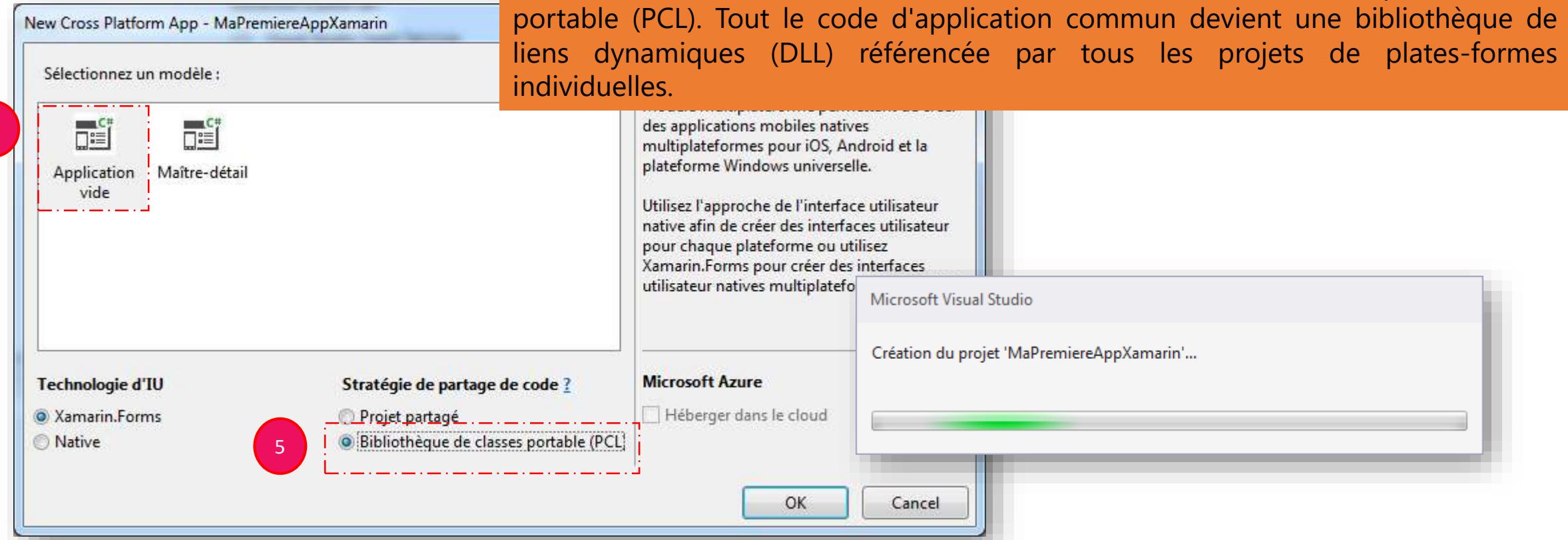
# Création d'un projet cross-plateforme



# Création d'un projet cross-plateforme

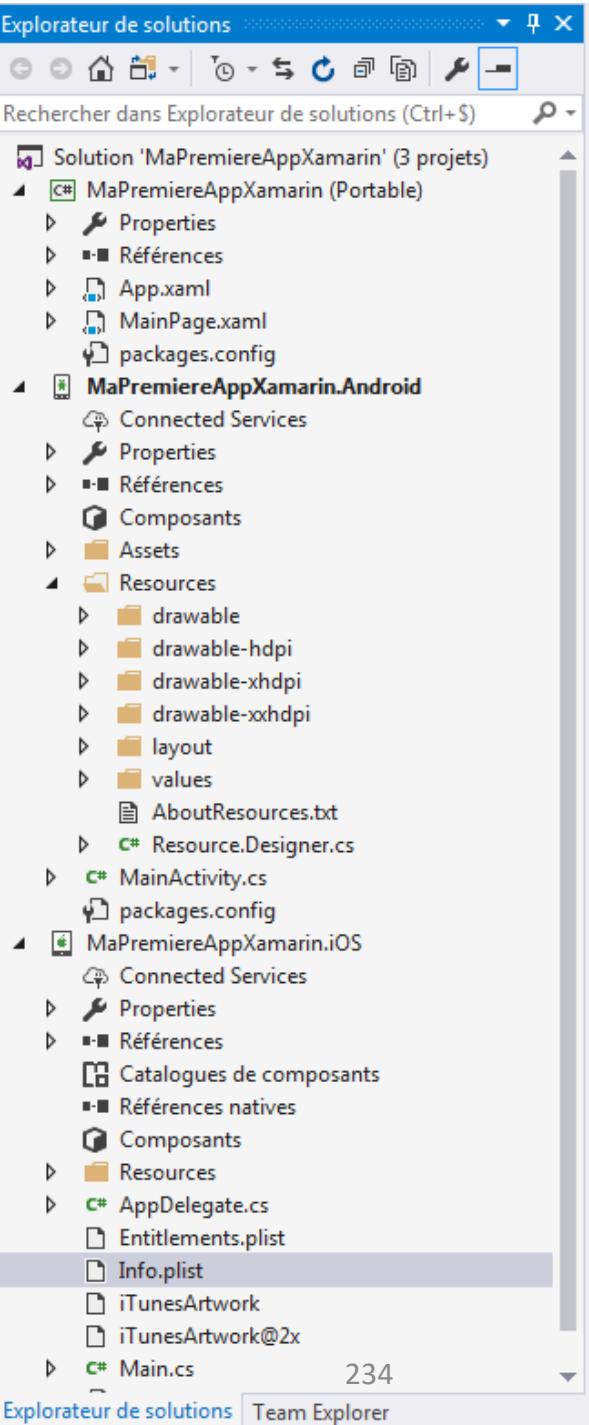
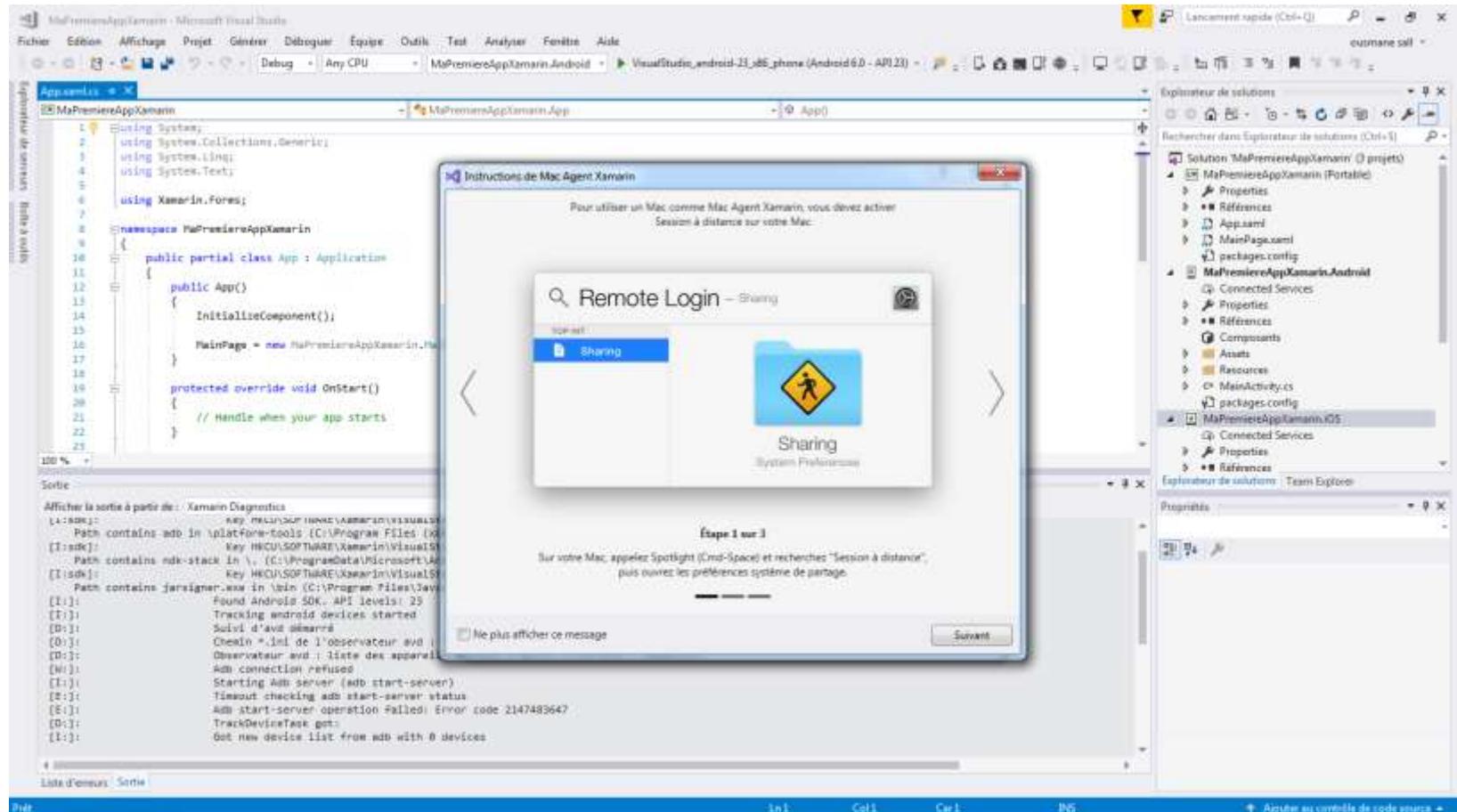


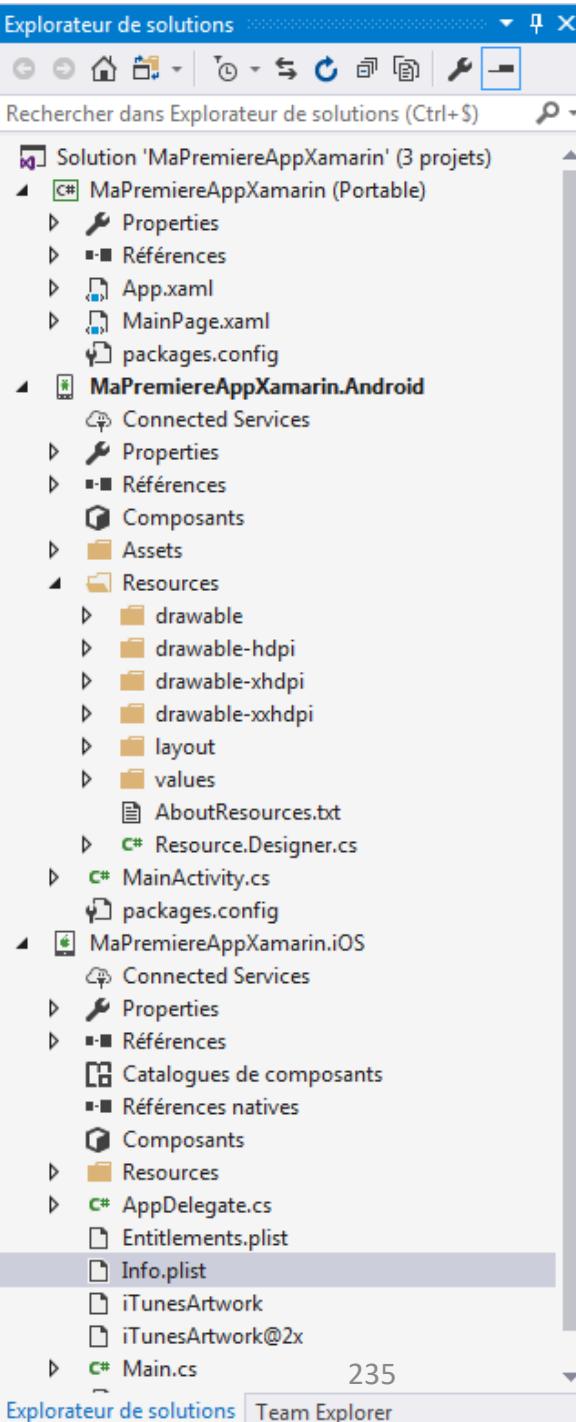
# Création d'un projet cross-plateforme



Le terme "partagé" dans ce contexte signifie Shared Asset Project (SAP) contenant des fichiers de code (et peut-être d'autres fichiers) qui sont partagés entre les projets de plate-forme, devenant essentiellement une partie de chaque projet de plate-forme.

# Création d'un projet cross-plateforme

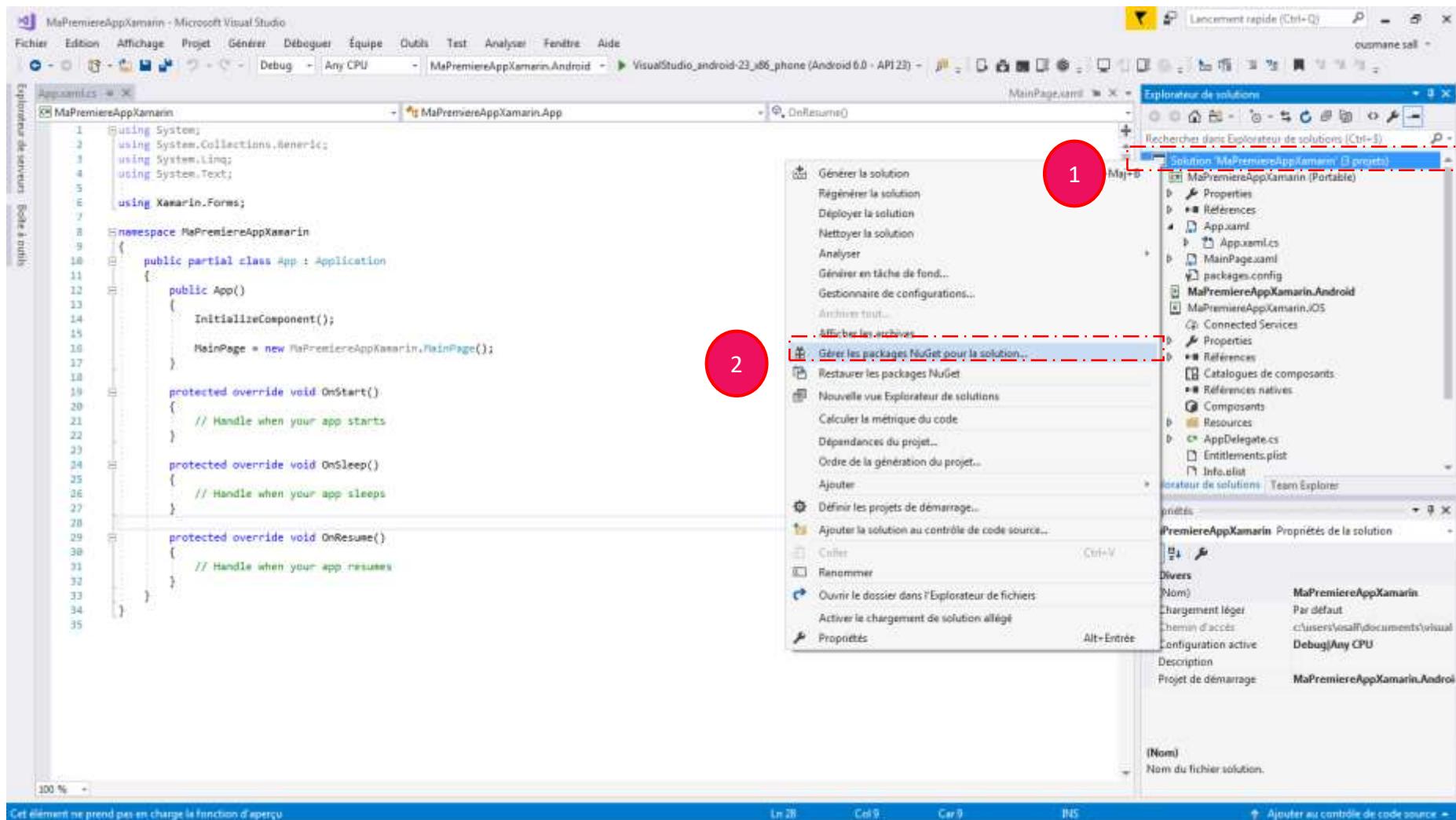




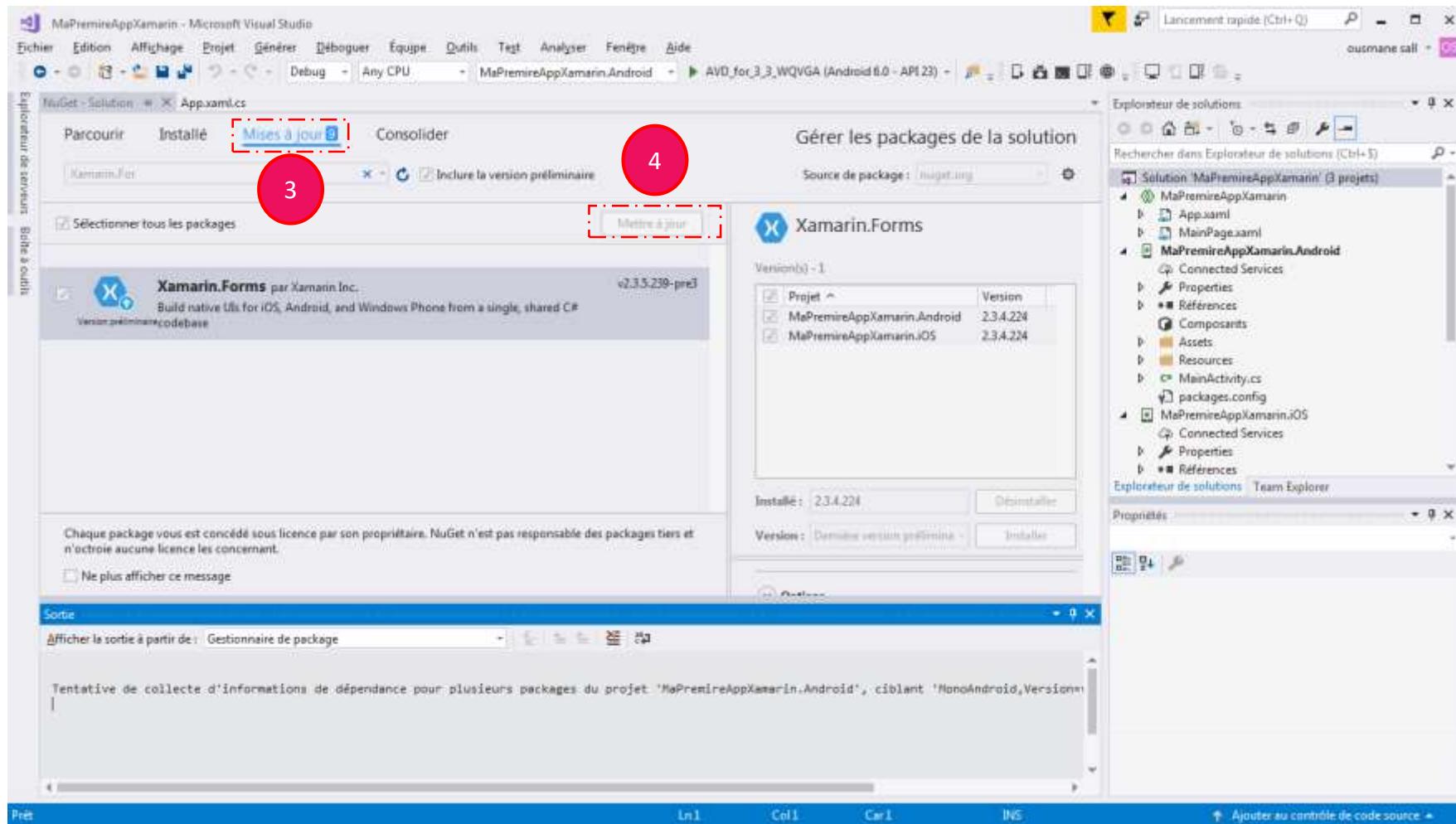
# Création d'un projet cross-plateforme

- Dans Visual Studio, trois projets au moins sont créés: un projet commun (le projet PCL) et deux projets d'application. Pour une solution nommée **MaPremiereAppXamarin**, ce sont :
  - A Portable Class Library project named **MaPremiereAppXamarin** that is referenced by all application projects;
  - An application project for Android, named **MaPremiereAppXamarin.Android**;
  - An application project for iOS, named **MaPremiereAppXamarin.iOS**.

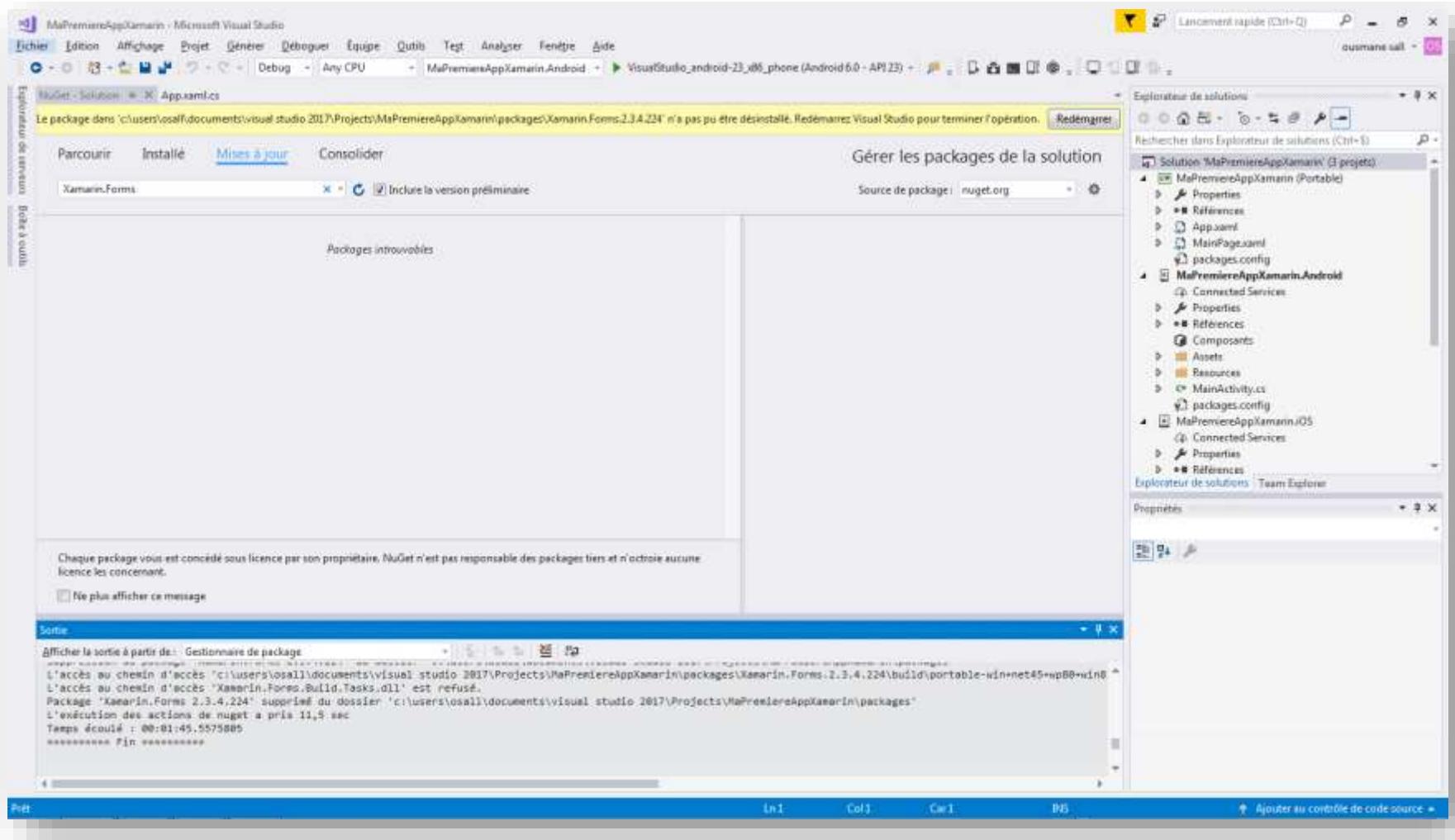
# Création d'un projet cross-plateforme



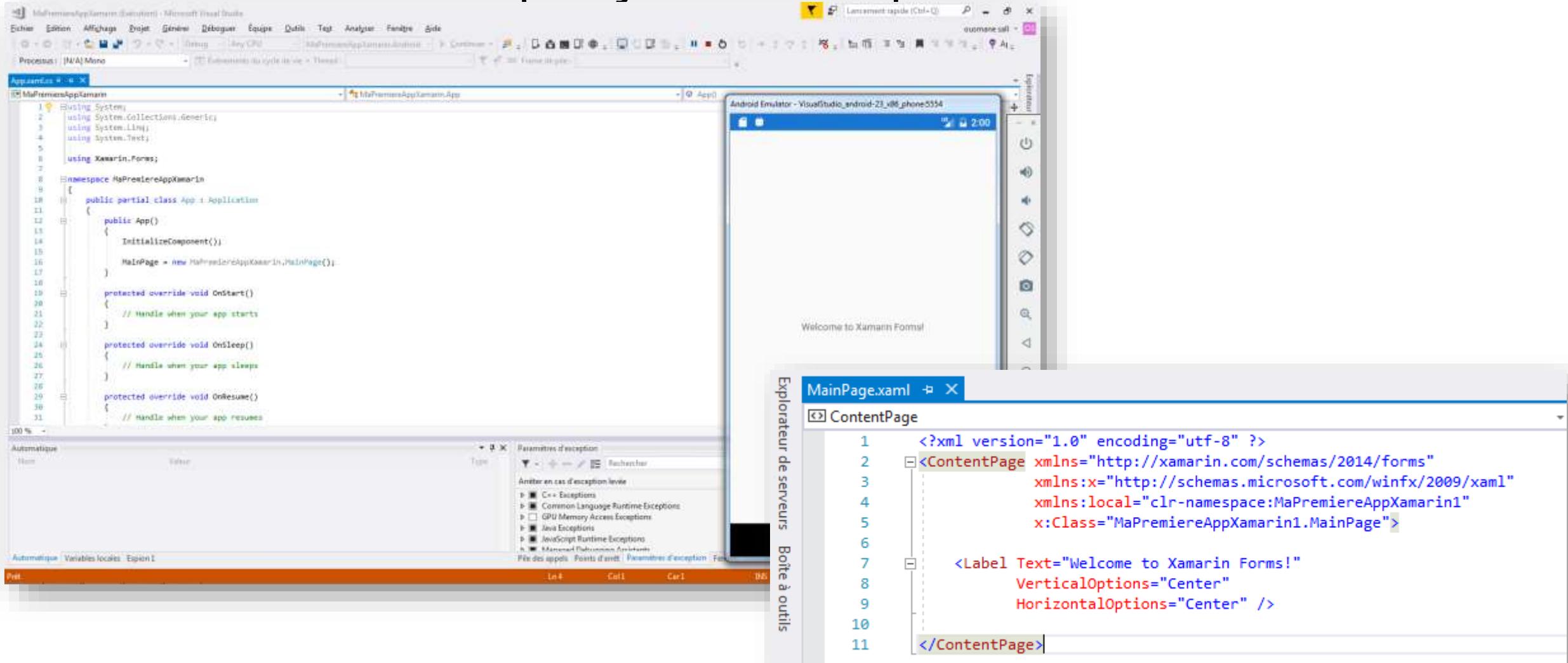
# Création d'un projet cross-plateforme



# Création d'un projet cross-plateforme



# Création d'un projet cross-plateforme

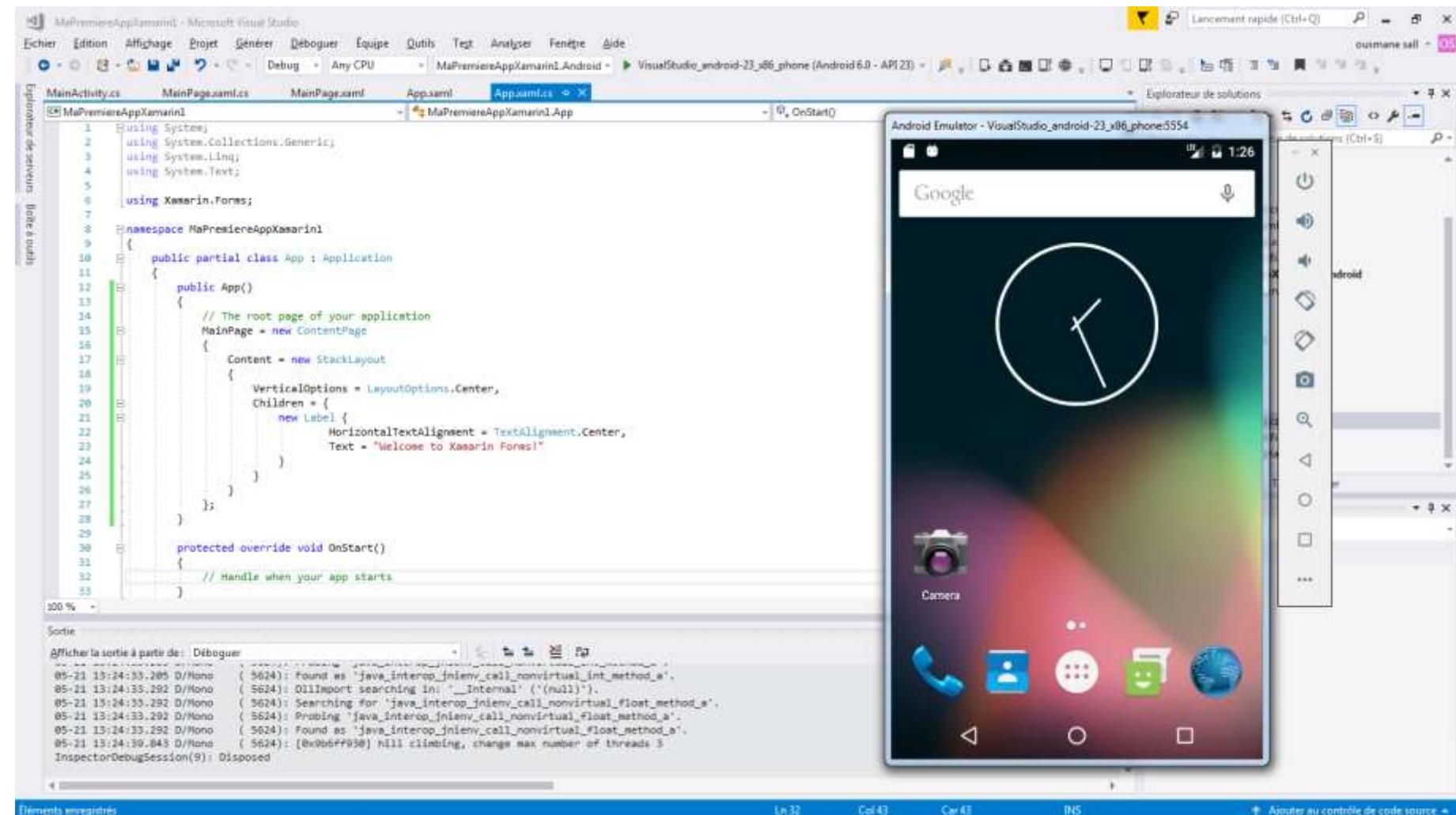


# Création d'un projet cross-plateforme :C#

Notez que l'espace de noms est identique au nom du projet. Cette App est définie comme publique et dérive de la classe d'application **Xamarin.Forms**. Le constructeur a une seule responsabilité: définir la propriété **MainPage** de la classe d'application à un objet de type **Page**.

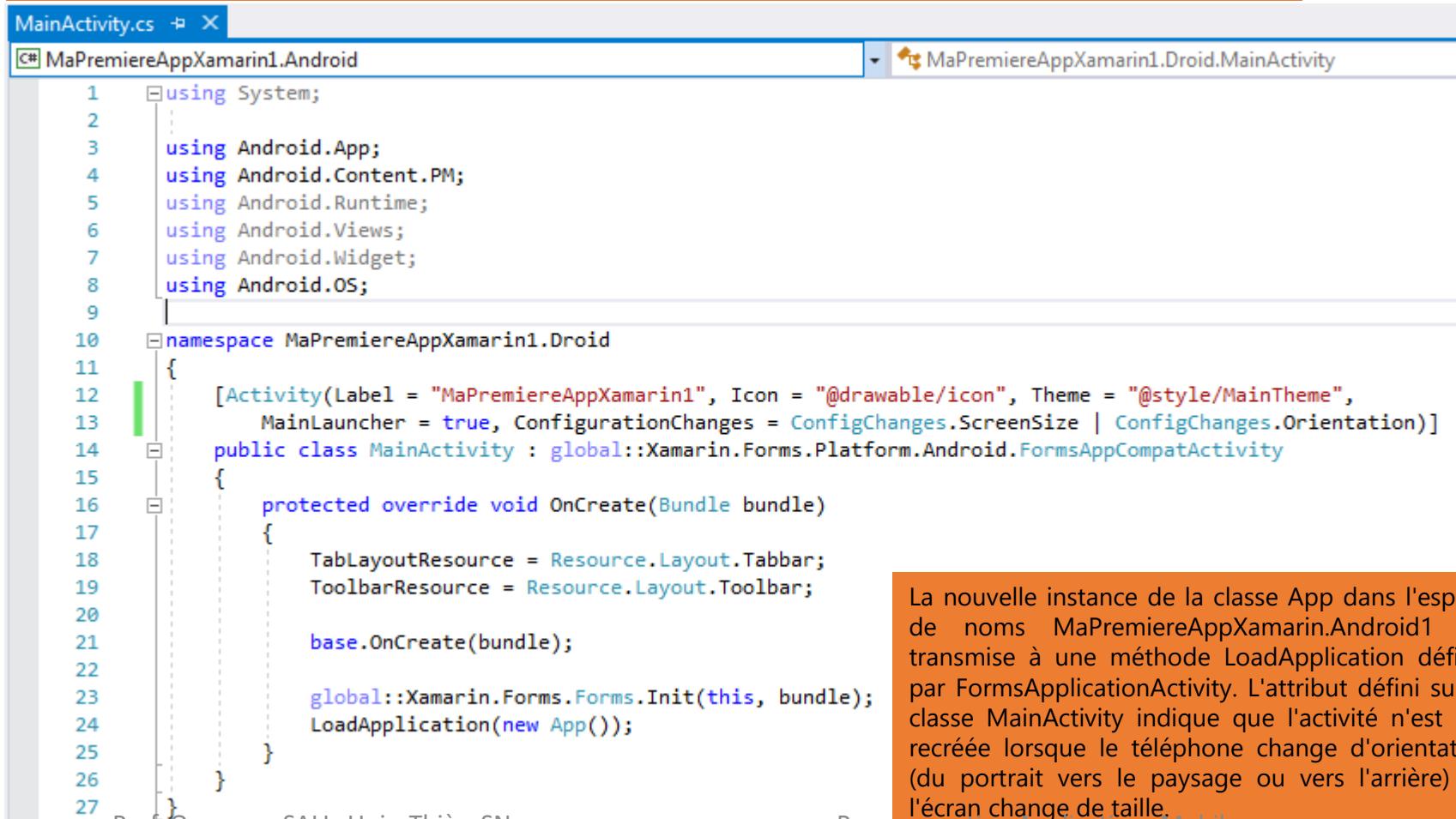
La classe **ContentPage** définit une propriété nommée **Content** que vous définissez sur le contenu de la page. Généralement, ce contenu est un layout qui, à son tour, contient un tas de vues, et dans ce cas, il est défini sur un **StackLayout**, qui range ses enfants dans une pile.

Ce **StackLayout** n'a qu'un seul enfant, qui est un label. La classe **Label** dérive de **View** et est utilisée dans les applications **Xamarin.Forms** pour afficher un paragraphe de texte. Les propriétés **VerticalOptions** et **"HorizontalTextAlignment"** sont traitées plus en détail plus loin dans ce cours.



# Le projet Android

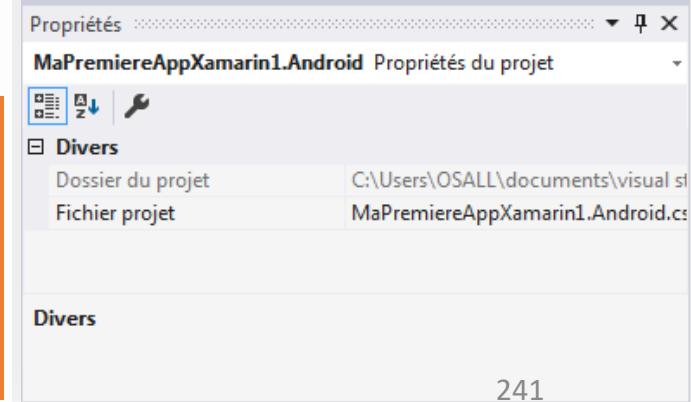
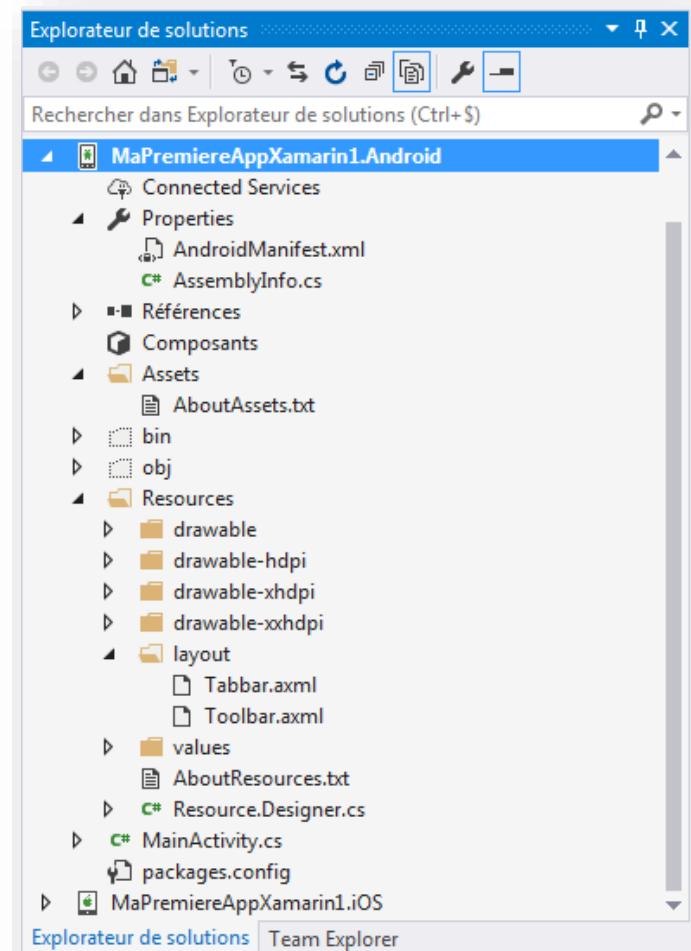
Dans l'application Android, la classe **MainActivity** typique doit être dérivée d'une classe **Xamarin.Forms** nommée **FormsApplicationActivity**, définie dans **Xamarin.Forms.Platform.Android** et l'appel de **Forms.Init** nécessite des informations supplémentaires:



The screenshot shows the Visual Studio IDE interface. On the left, the code editor displays the `MainActivity.cs` file for the `MaPremiereAppXamarin1.Android` project. The code defines a `MainActivity` class that inherits from `FormsAppCompatActivity`. It overrides the `OnCreate` method to set the tab bar and toolbar resources, and calls `Forms.Init` before loading the application. On the right, the Solution Explorer shows the project structure for `MaPremiereAppXamarin1.Android`, including files like `AndroidManifest.xml`, `AssemblyInfo.cs`, and various resource folders.

```
1  using System;
2
3  using Android.App;
4  using Android.Content.PM;
5  using Android.Runtime;
6  using Android.Views;
7  using Android.Widget;
8  using Android.OS;
9
10 namespace MaPremiereAppXamarin1.Droid
11 {
12     [Activity(Label = "MaPremiereAppXamarin1", Icon = "@drawable/icon", Theme = "@style/MainTheme",
13             MainLauncher = true, ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation)]
14     public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
15     {
16         protected override void OnCreate(Bundle bundle)
17         {
18             TabLayoutResource = Resource.Layout.Tabbar;
19             ToolbarResource = Resource.Layout.Toolbar;
20
21             base.OnCreate(bundle);
22
23             global::Xamarin.Forms.Forms.Init(this, bundle);
24             LoadApplication(new App());
25         }
26     }
27 }
```

La nouvelle instance de la classe App dans l'espace de noms `MaPremiereAppXamarin1` est transmise à une méthode `LoadApplication` définie par `FormsApplicationActivity`. L'attribut défini sur la classe `MainActivity` indique que l'activité n'est pas recréée lorsque le téléphone change d'orientation (du portrait vers le paysage ou vers l'arrière) ou l'écran change de taille.

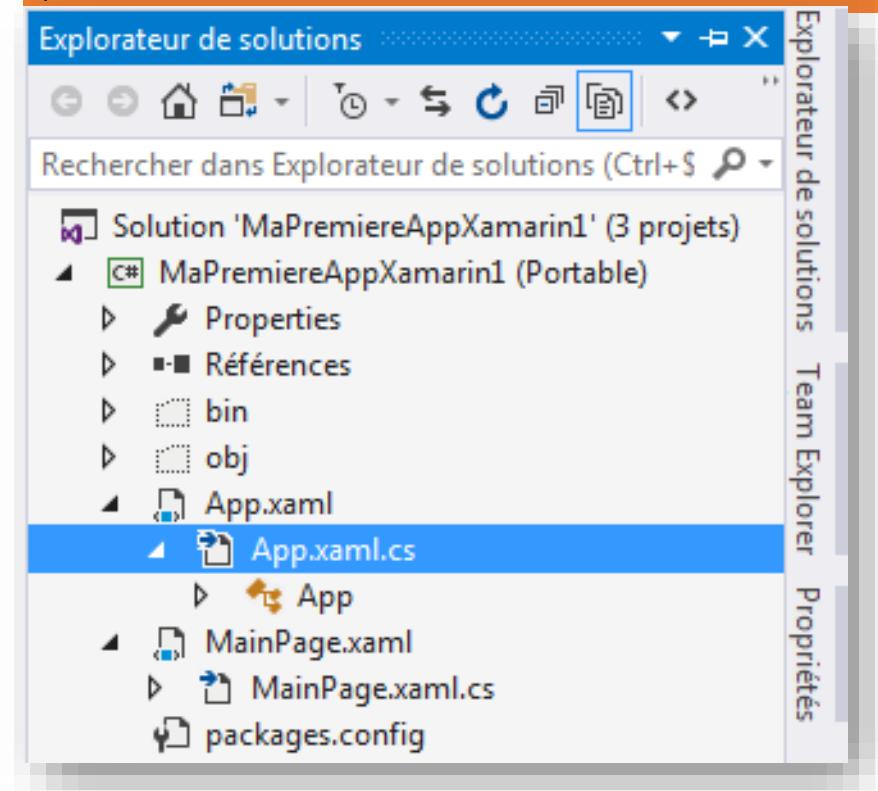


# Le projet portable

Maintenant, regardez le fichier **MainPage.xaml.cs** caché sous le fichier **MainPage.xaml** dans la liste des fichiers de projets.

Ce fichier définit la classe **MainPage** habituelle, mais elle provient en fait d'une classe `Xamarin.Forms` spécifiée comme élément racine dans le fichier **MainPage.xaml**.

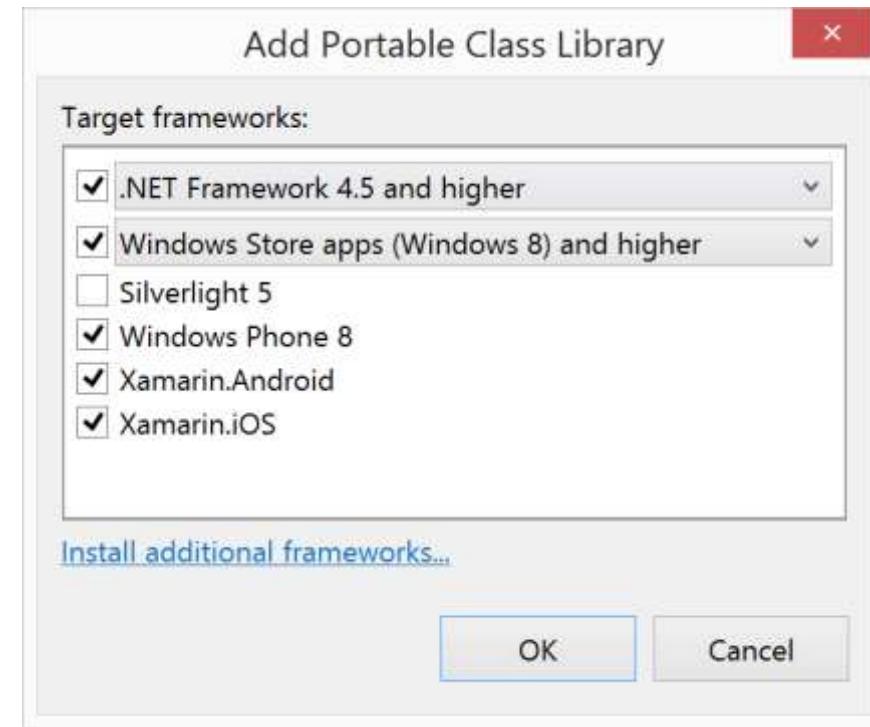
Une nouvelle classe d'application instanciée est passée à la méthode d'application de la ligne définie par cette classe de base:



```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5
6  using Xamarin.Forms;
7
8  namespace MaPremiereAppXamarin1
9  {
10    public partial class App : Application
11    {
12      public App()
13      {
14        // The root page of your application
15        InitializeComponent();
16        MainPage = new MaPremiereAppXamarin1.MainPage();
17      }
18
19      protected override void OnStart()
20      {
21        // Handle when your app starts
22      }
23
24      protected override void OnSleep()
25      {
26        // Handle when your app sleeps
27      }
28
29      protected override void OnResume()
30      {
31        // Handle when your app resumes
32      }
33    }
34  }
```

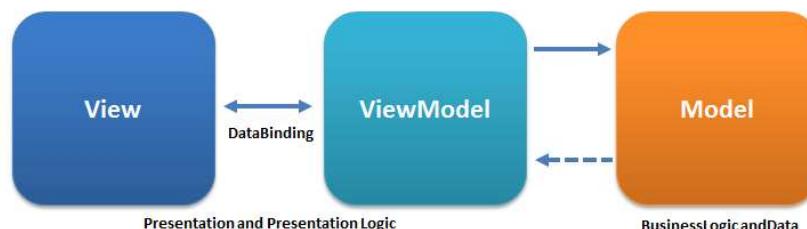
# Portable Class Libraries

- 1 Assembly
- Plusieurs plateformes
- Incluant :
  - **Xamarin.Android**
  - **Xamarin.iOS**



# MVC Design Pattern: Introduction du binding

- MVVM permet de lier les données à l'interface graphique. Sorte de pattern de développement qui permet de séparer la couche logique et l'IHM(sorte de MVC pour les applications XAML).
- Cette liaison s'effectue à l'aide d'un mécanisme appelé « Data-Binding »
- Cette liaison peut être:
  - De la donnée vers l'interface(Affichage dans un Label)
  - De l'Interface vers la donnée(Affichage dans une zone de texte)
  - Les 2( donnée--> interface et interface-->donnée)



# Exercice 3

- Questions
  - Quels sont les différentes étapes pour créer un projet Cross-Platform sous Xamarin ?
  - Rappelez le rôle des différents blocs Visual Studio ?
  - Quels sont les principaux types de fichiers qu'on peut trouver dans une application Cross-Platform ?

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# **INF 5312 Technologies Mobiles:**

## **Développement d'Applications**

## **Mobiles Cross-Platform avec Xamarin**

## **et C#**

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

**Xamarin - Développement Cross-Platform avec  
Xamarin.Forms**

# A propos de moi

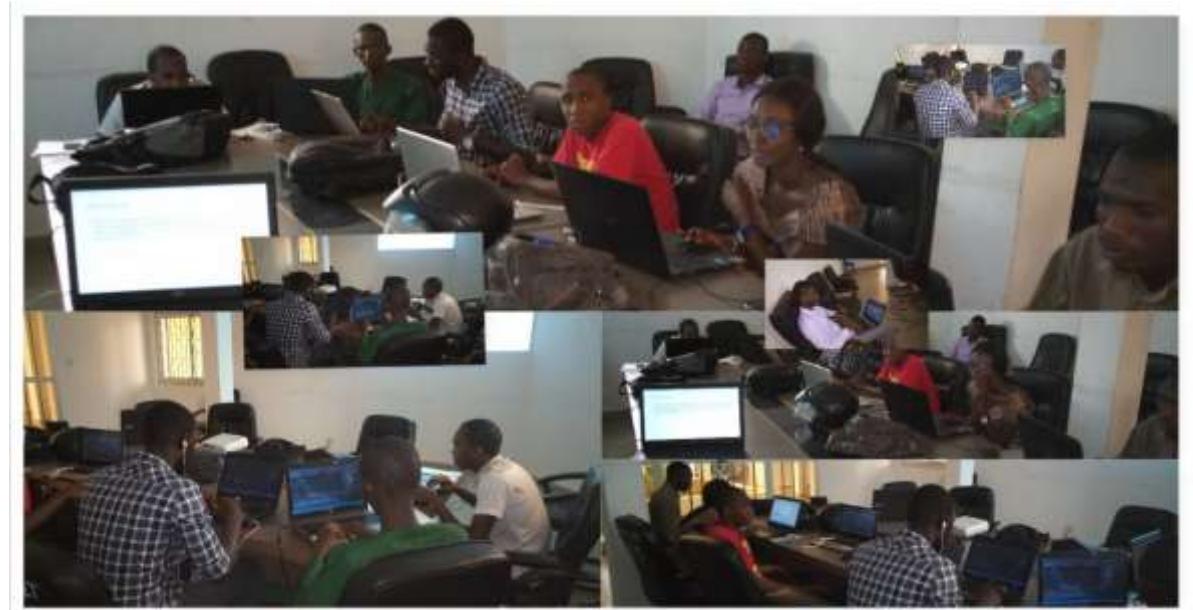


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

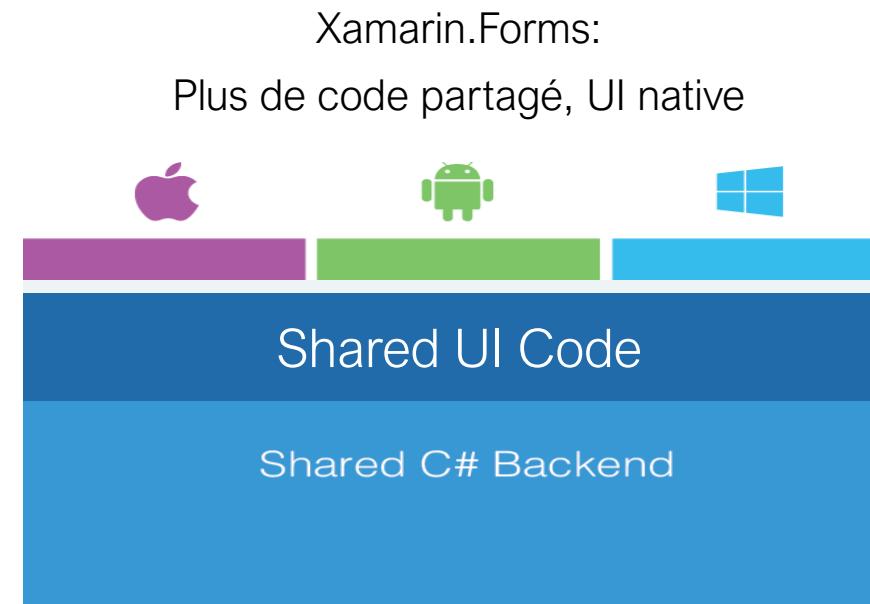
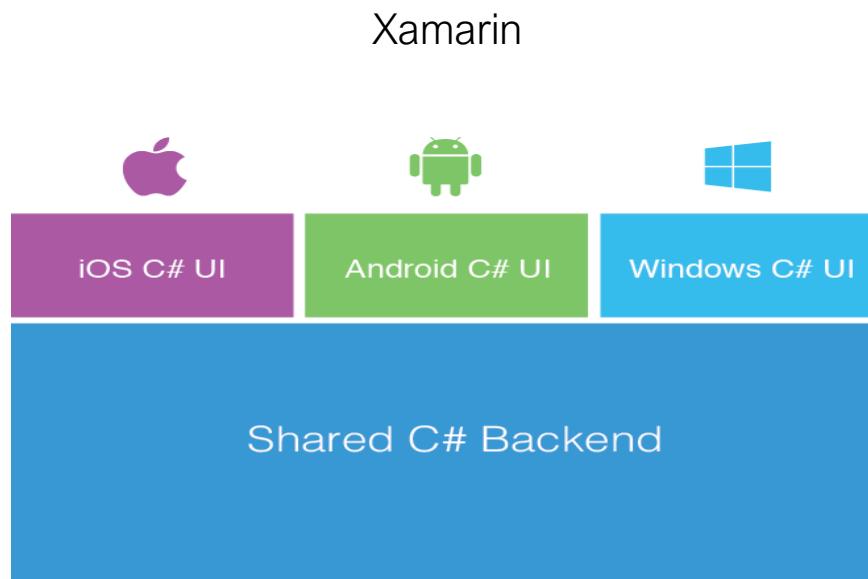
*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Xamarin pour le développement d'Applications Mobiles
3. Une architecture pour le développement Cross-Plateforme
- 4. Développement Cross-Platform avec Xamarin.Forms**
5. Interface utilisateur
6. Data-Binding
7. Web Services
8. Test et Déploiement

# L'approche de Xamarin.Forms



**Xamarin.Forms** une bibliothèque de Xamarin pour créer une interface utilisateur multiplate-forme.

Permet d'être très productif, de partager un code, mais de créer une interface utilisateur sur chaque plate-forme et d'accéder aux API de la plate-forme.

Avec **Xamarin.Forms**, vous avez maintenant une bonne couche de code UI partagé, mais vous avez toujours accès aux API de la plate-forme.

# Xamarin.Forms

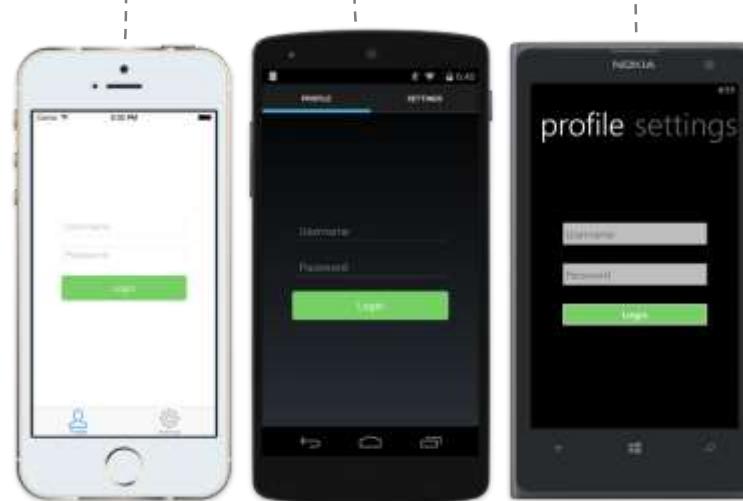
```
using Xamarin.Forms;

var profilePage = new ContentPage {
    Title = "Profile",
    Icon = "Profile.png",
    Content = new StackLayout {
        Spacing = 20, Padding = 50,
        VerticalOptions = LayoutOptions.Center,
        Children = {
            new Entry { Placeholder = "Username" },
            new Entry { Placeholder = "Password", IsPassword = true },
            new Button {
                Text = "Login",
                TextColor = Color.White,
                BackgroundColor = Color.FromHex("77D065") }}}
};

var settingsPage = new ContentPage {
    Title = "Settings",
    Icon = "Settings.png",
    (...)}
};

var mainPage = new TabbedPage { Children = { profilePage, settingsPage } };
```

À l'exécution, les pages et contrôles Xamarin.Forms sont transformés en composants natifs.



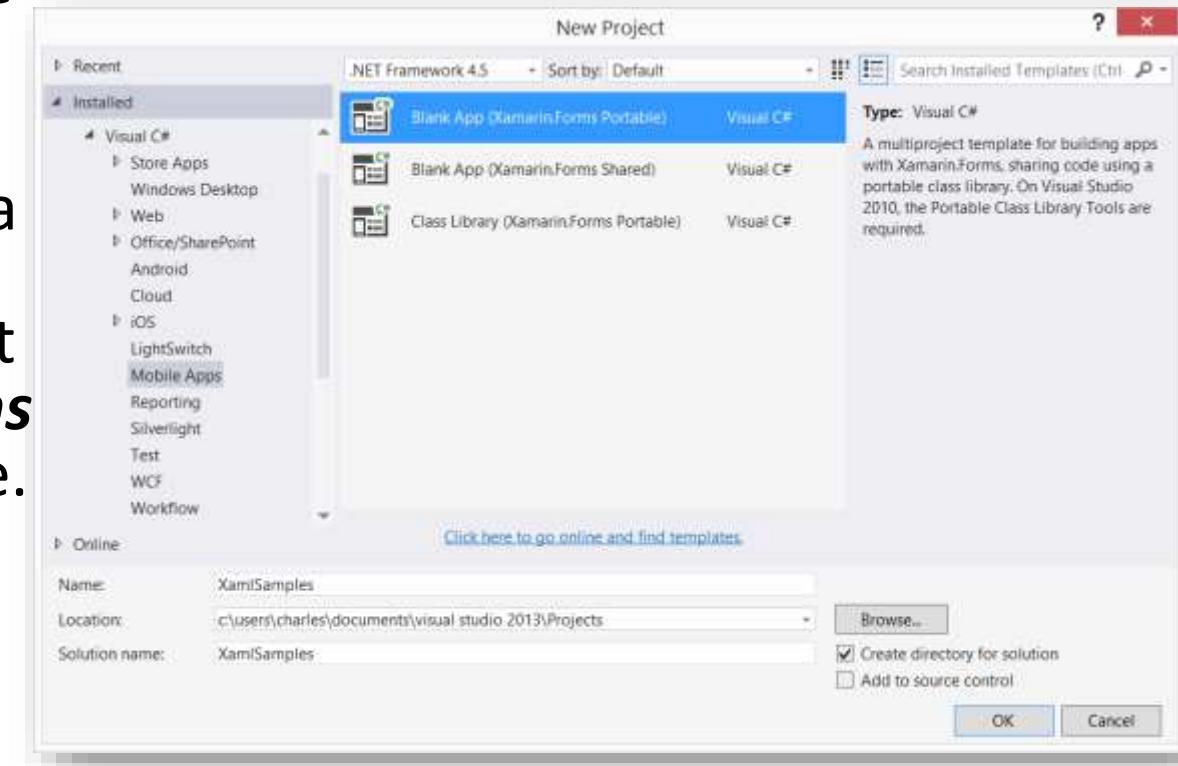
Utilisez une seule API pour générer une interface native pour chaque plateforme

# eXtensible Application Markup Language (XAML)

- XAML est un langage de balisage déclaratif qui peut être utilisé pour définir les interfaces utilisateurs. L'interface utilisateur est définie dans un fichier XML à l'aide de la syntaxe XAML, tandis que le comportement d'exécution est défini dans un fichier distinct derrière le code (\*.cs).
- Dans une application Xamarin.Forms, XAML est principalement utilisé pour définir le contenu visuel d'une page.
  - Un fichier XAML est toujours associé à un fichier de code C # qui fournit un support de code pour le balisage.
  - Ensemble, ces deux fichiers comprennent une nouvelle définition de classe qui inclut les vues pour enfants et l'initialisation des propriétés.
  - Dans le fichier XAML, les classes et les propriétés sont référencées avec des éléments et des attributs XML, et les liens entre le balisage et le code sont établis.

# eXtensible Application Markup Language (XAML)

- Pour commencer à éditer votre premier fichier XAML, utilisez Visual Studio ou Xamarin Studio pour créer une nouvelle solution Xamarin.Forms.
- Dans Visual Studio, sélectionner **File > New > Project** à partir du menu. Dans la boîte de dialogue *New Project*, sélectionner **Visual C# > Mobile Apps** à gauche, et **Blank App (Xamarin.Forms Portable)** au niveau de la liste au centre.
- Cela créera une solution basée PCL qui supporte XAML.

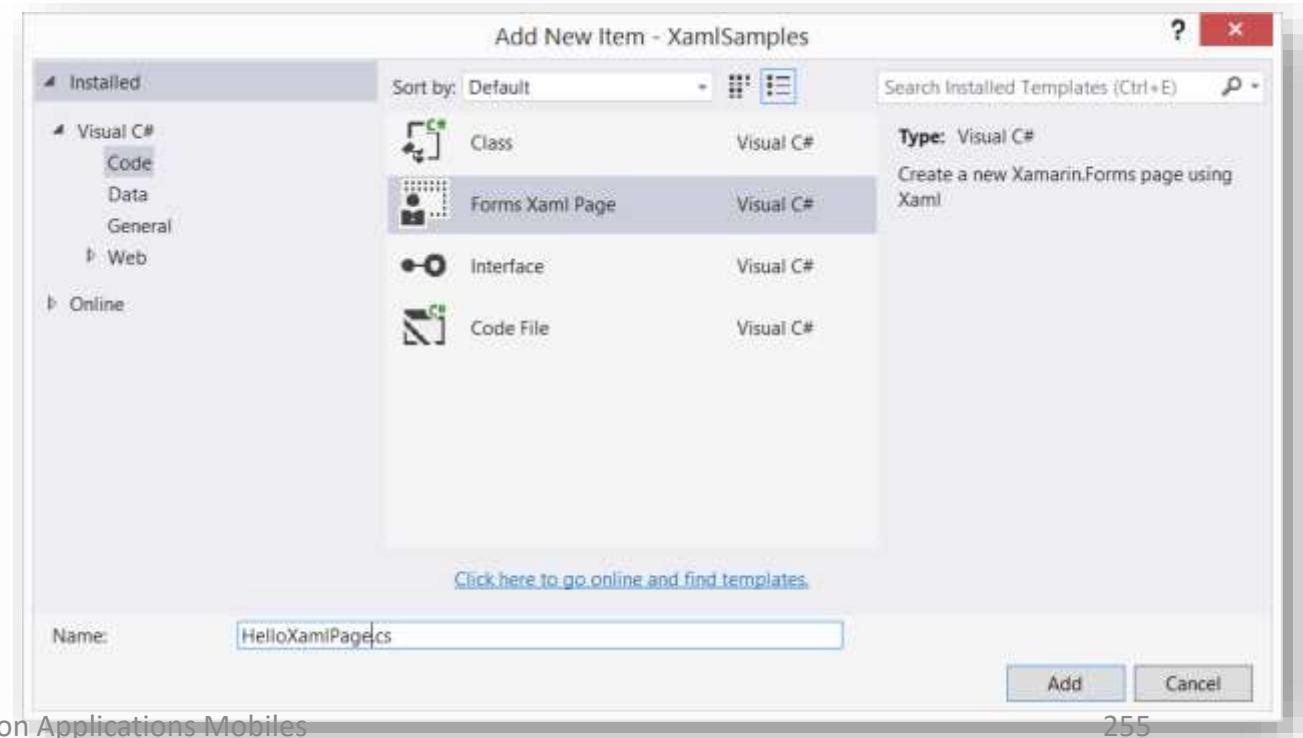


# eXtensible Application Markup Language (XAML)

- Dans Visual Studio, click bouton droit sur le projet XamlSamples et selectionner *Add > New Item*. Dans la boîte *Add New Item*, selectionner *Visual C# > Code* à gauche, et *Forms Xaml Page* sur la liste.

Deux nouveaux fichiers sont créés dans le projet XamlSamples: le premier est un fichier XAML nommé **HelloXamlPage.xaml**. Le deuxième fichier s'affiche en retrait; Ceci est un fichier de code C # avec le nom inhabituel **HelloXamlPage.xaml.cs**. Les noms de ces deux fichiers révèlent qu'ils sont intimement liés. Le fichier C # est souvent appelé le fichier code-behind du fichier XAML.

Les deux **HelloXamlPage.xaml** et **HelloXamlPage.xaml.cs** contribuent à la définition d'une classe nommée **HelloXamlPage** dérivée de **ContentPage**.



# Anatomie d'une classe XAML

- Dans **HelloXamlPage.xaml**, la première chose que vous devez faire est de supprimer tout ce qui apparaît entre les étiquettes de début et de fin afin que le fichier ressemble à ceci:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="XamlSamples.HelloXamlPage">
</ContentPage>
```

- L'attribut **x: Class** spécifie un nom de classe .NET : la classe **HelloXamlPage** dans l'espace de noms **XamlSamples**. Cela signifie que ce fichier XAML définit une nouvelle classe nommée **HelloXamlPage** dans l'espace de noms **XamlSamples** qui découle de **ContentPage**: l'étiquette dans laquelle apparaît l'attribut x: Class.

# HelloXamlPage.xaml.cs code-behind

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace XamlSamples
{
    public partial class HelloXamlPage
    {
        public HelloXamlPage()
        {
            InitializeComponent();
        }
    }
}
```

# eXtensible Application Markup Language (XAML)

- Recherchez la classe App dans le projet XamlSamples, vous pouvez supprimer certains des codes existants et utiliser le constructeur de l'application pour configurer MainPage dans une instance de HelloXamlPage:

```
namespace XamlSamples
{
    public class App : Xamarin.Forms.Application
    {
        public App ()
        {
            MainPage = new HelloXamlPage();
        }
    }
}
```

- Le projet peut maintenant être compilé pour l'une des trois plates-formes, mais la page est entièrement vide.

# Définition du contenu de la Page

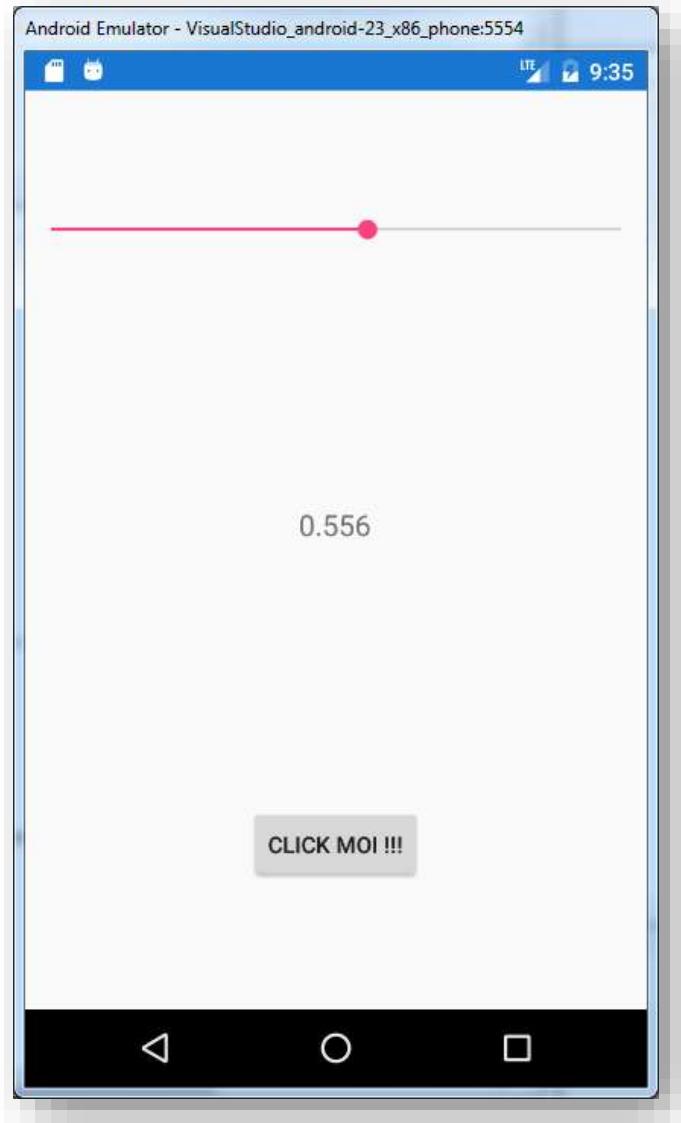
- Une application Xamarin.Forms d'une seule page contient généralement une classe dérivée de ContentPage. La propriété Content de cette classe est généralement définie sur une vue unique ou une mise en page avec des vues enfants. Dans XAML, une vue (par exemple une étiquette) peut être implicitement définie dans la propriété Content de la page en étant placé entre les tags ContentPage de début et de fin

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="XamlSamples.HelloXamlPage"
    Title="Hello XAML Page"
    Padding="10, 40, 10, 10">

    <Label Text="Hello, XAML!"
        VerticalOptions="Start"
        HorizontalTextAlignment="Center"
        Rotation="-15"
        IsVisible="true"
        FontSize="Large"
        FontAttributes="Bold"
        TextColor="Aqua" />

</ContentPage>
```

# XAML et Interactions avec le Code



# XAML et Interactions avec le Code

L'exemple HelloXamlPage contient uniquement un seul `Label` sur la page, mais c'est très inhabituel. La plupart des dérivés `ContentPage` définissent la propriété `Content` dans une mise en page d'une sorte telle qu'un `StackLayout`. La propriété `Enfants` de `StackLayout` est définie comme étant de type `IList<View>`, mais il s'agit en fait d'un objet de type `ElementCollection<View>`, et cette collection peut être remplie avec plusieurs vues ou autres mises en page. Dans XAML, ces relations parent-enfant sont établies avec une hiérarchie XML normale. Voici un fichier XAML pour une classe nommée `XamlPlusCodePage`:

The screenshot shows two code editors side-by-side. The left editor, titled "MainPage.xaml", contains XAML code defining a ContentPage with a StackLayout containing a Slider and a Button, and a Label with a ValueChanged event handler. The right editor, titled "MainPage.xaml.cs", contains C# code for the MainPage class, which includes event handlers for the slider's ValueChanged event and the button's Clicked event.

```
MainPage.xaml
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      xmlns:local="clr-namespace:HelloXamlPage"
5      x:Class="HelloXamlPage.MainPage"
6      Title="Exemple de XAML et Interactions avec le Code">
7      <StackLayout>
8          <Slider VerticalOptions="CenterAndExpand"
9              ValueChanged="OnSliderValueChanged"/>
10         <Label x:Name="valueLabel"
11             Text="Un simple Label"
12             Font="Large"
13             VerticalOptions="CenterAndExpand"
14             HorizontalOptions="Center" />
15
16         <Button Text="Click moi !!!"
17             HorizontalOptions="Center"
18             VerticalOptions="CenterAndExpand"
19             Clicked="OnButtonClicked"/>
20     </StackLayout>
21 </ContentPage>

MainPage.xaml.cs
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using Xamarin.Forms;
7
8  namespace HelloXamlPage
9  {
10     public partial class MainPage : ContentPage
11     {
12         public MainPage()
13         {
14             InitializeComponent();
15         }
16
17         private void OnSliderValueChanged(object sender, ValueChangedEventArgs e)
18         {
19             valueLabel.Text = ((Slider)sender).Value.ToString("F3");
20         }
21
22         async void OnButtonClicked(object sender, EventArgs e)
23         {
24             Button monBouton = (Button)sender;
25             await DisplayAlert("Click", "Le bouton avec le label " + monBouton.Text + " a été cliqué", "OK");
26         }
27     }
28 }
```

# Cycle de vie de l'application

- La classe Application contient trois méthodes virtuelles qui peuvent être remplacées pour gérer les méthodes de cycle de vie:
  - OnStart - Appelé lorsque l'application est chargée.
  - OnSleep - Appelé chaque fois que l'application est en pause.
  - OnResume - Appelé lors de la reprise de l'application, après avoir été envoyé en pause.

```
protected override void OnStart()
{
    Debug.WriteLine ("OnStart");
}
protected override void OnSleep()
{
    Debug.WriteLine ("OnSleep");
}
protected override void OnResume()
{
    Debug.WriteLine ("OnResume");
}
```

# Afficher les fenêtres contextuelles: afficher des alertes et guider les utilisateurs à travers des tâches

- **Xamarin.Forms** fournit deux types de pop-up comme élément de l'interface utilisateur – une boîte alerte et une “**action sheet**”. Nous allons voir comment utiliser les API d'alerte et de « **feuille d'action** » pour poser aux utilisateurs des questions simples et pour guider les utilisateurs à travers des tâches.
- L'affichage d'une alerte ou la demande à un utilisateur de faire un choix est une tâche assez commune d'une UI. **Xamarin.Forms** dispose de deux méthodes sur la classe de pages pour interagir avec l'utilisateur via un pop-up: **DisplayAlert** et **DisplayActionSheet**. Elles sont rendus avec des contrôles natifs appropriés sur chaque plate-forme.

# Affichage d'une Alert

- Toutes les plates-formes supportées par **Xamarin.Forms** ont une pop-up modal pour alerter l'utilisateur ou poser des questions simples. Pour afficher ces alertes dans **Xamarin.Forms**, utilisez la méthode **DisplayAlert** sur n'importe quelle page. La ligne de code suivante montre un message simple à l'utilisateur:

The screenshot shows two code editors side-by-side. The top editor is titled "MainPage.xaml" and displays XAML code for a ContentPage. The bottom editor is titled "MainPage.xaml.cs" and displays C# code for the MainPage class.

**MainPage.xaml:**

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:local="clr-namespace:AlertProject"
    x:Class="AlertProject.MainPage">

    <Button x:Name="monBouton"
        Text="Click Me !!!"
        VerticalOptions="Center"
        HorizontalOptions="Center"
        BackgroundColor="Green"
        Clicked="monBouton_Clicked"/>

</ContentPage>
```

**MainPage.xaml.cs:**

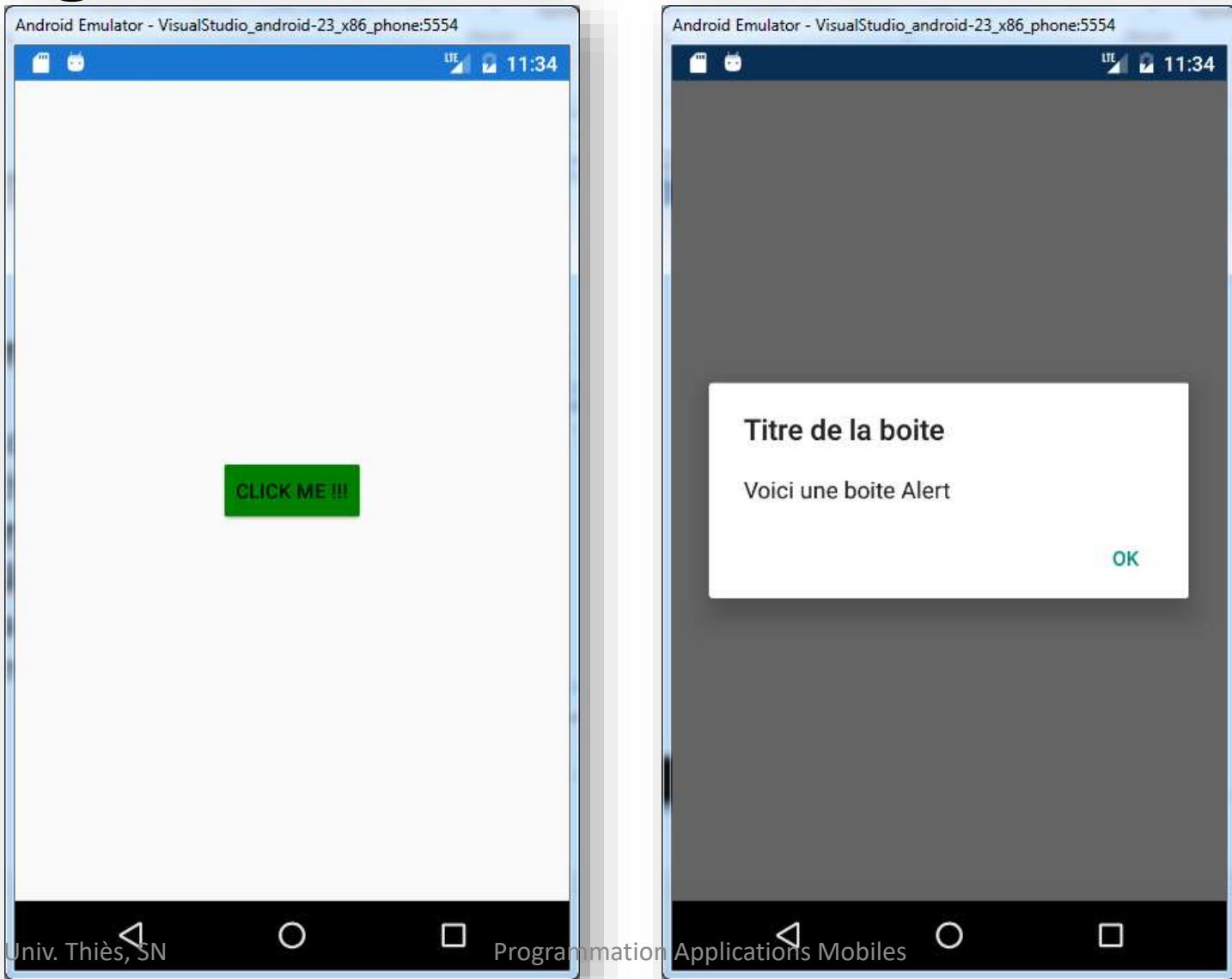
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Xamarin.Forms;

namespace AlertProject
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

        private void monBouton_Clicked(object sender, EventArgs e)
        {
            DisplayAlert("Titre de la boite", "Voici une boite Alert", "OK");
        }
    }
}
```

A red dashed rectangle highlights the call to `DisplayAlert` in the C# code.

# Affichage d'une Alert



# Affichage d'une Alert: récupérer la réponse d'un utilisateur

- La méthode **DisplayAlert** peut également être utilisée pour récupérer la réponse d'un utilisateur en présentant deux boutons et en renvoyant un booléen.
- Pour obtenir la réponse d'une boîte alerte, fournissez le texte pour les deux boutons et attendez la méthode. Une fois que l'utilisateur sélectionne l'une des options, la réponse sera retournée à votre code.
- Notez les mots clés **async** et **await** dans l'exemple de code ci-dessous:

```
async void OnAlertYesNoClicked (object sender, EventArgs e)
{
    var answer = await DisplayAlert ("Question?", "Would you like to play a game", "Yes", "No");
    Debug.WriteLine ("Answer: " + answer);
}
```



# Guider les utilisateurs à travers les tâches

```
public Task<string> DisplayActionSheet(string title, string cancel, string destruction, params string[] buttons);
```

- Le fichier **UIActionSheet** est un élément UI commun dans iOS. La méthode Xamarin.Forms **DisplayActionSheet** vous permet d'inclure ce contrôle dans les applications cross-platesformes, de créer des alternatives natives dans Android et Windows Phone.
- Pour afficher une feuille d'action, mettez **DisplayActionSheet** dans n'importe quelle page, en passant les étiquettes des messages et des boutons en tant que chaînes. La méthode renvoie l'étiquette de chaîne du bouton qui a été cliqué par l'utilisateur. Un exemple simple est montré ici:

```
private async void Button_Clicked(object sender, EventArgs e)
{
    var reponse = await DisplayActionSheet("Choisir votre Master",
        "Annuler",
        null,
        "Master IHA", "Master MA", "Master Inf");
    if(reponse != "Annuler")
        await DisplayAlert("Réponse", "Vous êtes en " + reponse, "Ok");
    else
        await DisplayAlert("Réponse", "Vous avez annulé !!!", "OK");
}
```

# DisplayActionSheet



# Exercice 4

- Créer un nouveau projet dont il faut changer la couleur de fond de la fenêtre principale en jaune et de changer l'icône avec une image de votre choix.
- Ajouter trois boutons portants les labels « rouge », « bleu » et « vert » permettant de changer la couleur de fond de la fenêtre principale en la couleur spécifiée par leur label.
- Ajouter un nouveau bouton permettant lorsque vous appuierez dessus de changer avec une couleur choisie au hasard celle d'une zone de texte. Une boîte alerte sera affichée avec le nom la couleur.



# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 5312 Technologies Mobiles: Développement d'Applications Mobiles Cross-Platform avec Xamarin et C#

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

Xamarin - Interface Utilisateur

# A propos de moi

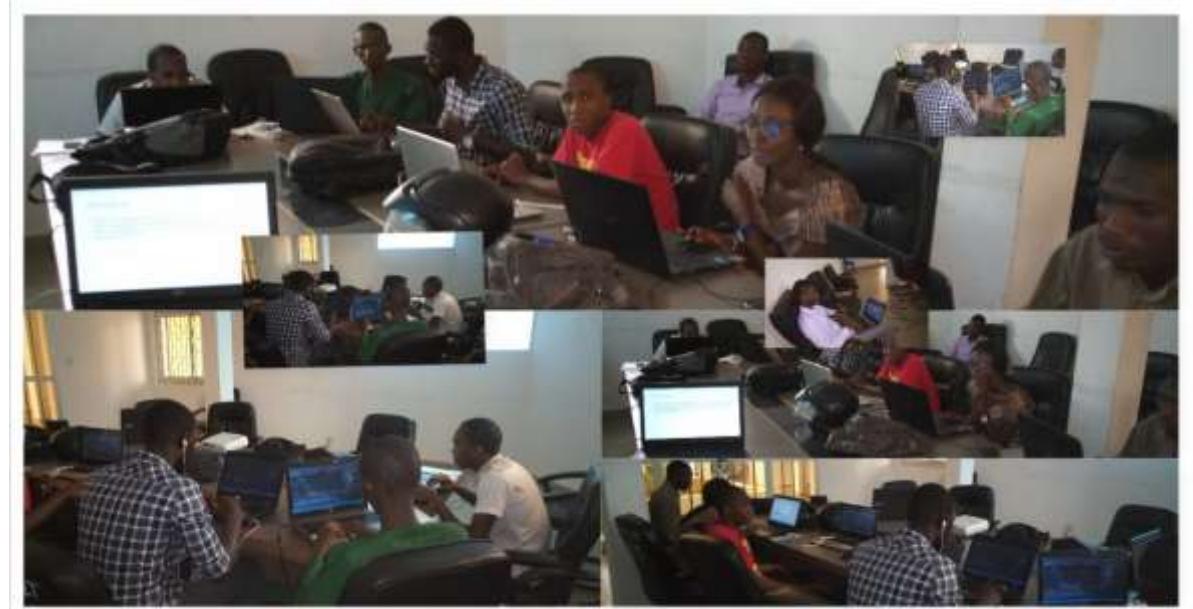


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Xamarin pour le développement d'Applications Mobiles
3. Une architecture pour le développement Cross-Plateforme
4. Développement Cross-Platform avec Xamarin.Forms
- 5. Interface utilisateur**
6. Data-Binding
7. Web Services
8. Test et Déploiement

# Propriétés communes des éléments visuels de l’UI Xamarin.Forms avec XAML

- Certaines propriétés communes à la plupart des contrôles, pages, layouts incluent :
  - **x:Name** : Le nom du View qui peut être utilisé pour faire référence à elle à partir du code
  - **AnchorX, AnchorY** : Contrôle des positions dans la mise en page
  - **HeightRequest, WidthRequest** : La taille demandée du View si le layout le permet
  - **MinimumHeightRequest, MinimumWidthRequest** : Taille minimale
  - **BackgroundColor** : Couleur de fond
  - **Rotation, RotationX, RotationY** : Propriétés de rotation
  - **Scale** : Valeur de mise en échelle(Agrandissement, rétrécissement)

# Propriétés communes des éléments visuels de l'UI Xamarin.Forms avec XAML

- Déterminer la place d'un élément dans un espace donné se fait par les attributs **VerticalOptions** et **HorizontalOptions**. Ceux-ci acceptent huit valeurs basées sur quatre comportements décrits comme suit :
  - **Start** : L'élément sera placé au plus haut possible dans l'espace donné par son conteneur parent
  - **Center** : L'élément sera placé au centre
  - **End** : L'élément sera placé en bas
  - **Fill** : L'élément prendra à partir de sa position prévue le maximum de place possible jusqu'au prochain élément

# Propriétés communes des éléments visuels de l'UI Xamarin.Forms avec XAML

- Les quatre comportements(Start, Center, End, Fill) peuvent être agrémentés d'une **stratégie d'expansion s'il reste de la place**. Dans ce cas, la place restante sera divisée entre tous les éléments disposant de cette stratégie d'expansion, et l'élément sera placé à l'intérieur. Pour assigner cette stratégie, il faudra utiliser les quatre autres valeurs :
  - **StartAndExpand**
  - **CenterAndExpand**
  - **EndAndExpand**
  - **FillAndExpand**

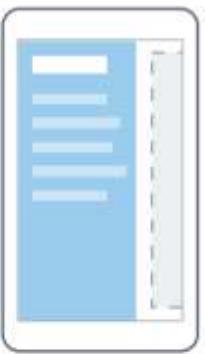
# Vue d'ensemble de l'architecture des interfaces: Pages

# Pages

- La classe page est le conteneur primaire de chaque écran principal dans une application. Dérivée de **Xamarin.forms.VisualElement**, une page est la classe de base pour la création d'autres classes de l'interface utilisateur de haut niveau. Voici les pages primaires :



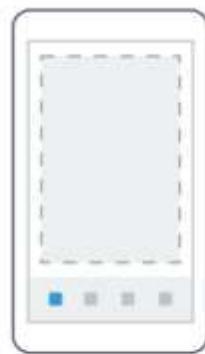
ContentPage



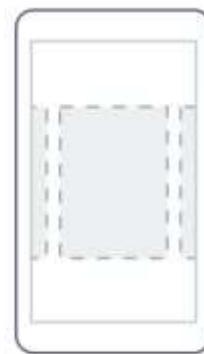
MasterDetailPage



NavigationPage



TabbedPage



CarouselPage

La classe page est le conteneur primaire de chaque écran principal dans une application. Dérivée de **Xamarin.forms.VisualElement**, une page est la classe de base pour la création d'autres classes de l'interface utilisateur de haut niveau.

# Titre des pages

- Les applications disposent d'un en-tête pour leur page, présentant le titre de celle-ci. Pour satisfaire cette contrainte, Xamarin propose un attribut **Title** sur les contrôles de type Page.

```
<ContentPage Title="Titre de ma page" ...>  
    <!-- Contenu omis -->  
</ContentPage>
```

A [ContentPage](#) displays a single [View](#), often a container such as a [StackLayout](#) or a [ScrollView](#).

# ContentPage

Il s'agit d'une page affichant une seule vue, souvent un conteneur comme **StackLayout** ou **ScrollView**.

L'exemple ci-dessous est pris à partir du fichier **App.cs**. Il utilise un **ContentPage** pour afficher une étiquette, qui est typique, mais basique, de la classe **ContentPage**.

```
MainPage.xaml.cs
```

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:ExempleTabbedPage"
             x:Class="ExempleTabbedPage.MainPage"
             Title="Page principale">
    <Label Text="Welcome to Xamarin Forms!"
          VerticalOptions="Center"
          HorizontalOptions="Center" />
</ContentPage>
```

```
using System;
using Xamarin.Forms;

namespace ContentPageExample
{
    public class App
    {
        public static Page Get MainPage ()
        {
            return new ContentPage {
                Content = new Label {
                    Text = "Hello, Forms!",
                    VerticalOptions = LayoutOptions.CenterAndExpand,
                    HorizontalOptions = LayoutOptions.CenterAndExpand,
                },
            };
        }
    }
}
```



# MasterDetailPage pour mettre un menu



## Public Fields

static readonly	<a href="#">IsGestureEnabledProperty</a>	<a href="#">BindableProperty</a> . Backing store for the <code>IsGestureEnabled</code> bindable property.
static readonly	<a href="#">IsPresentedProperty</a>	<a href="#">BindableProperty</a> . Backing store for the <code>MasterDetailPage.IsPresented</code> property.
static readonly	<a href="#">MasterBehaviorProperty</a>	<a href="#">BindableProperty</a> . Backing store for the <code>MasterBehavior</code> property.

## Public Properties

<a href="#">Detail</a>	<a href="#">Page</a> . Gets or sets the detail page that is used to display details about items on the master page.
<a href="#">IsGestureEnabled</a>	<a href="#">Boolean</a> . Gets or sets a value that turns on or off the gesture to reveal the master page. This is a bindable property.
<a href="#">IsPresented</a>	<a href="#">Boolean</a> . Gets or sets a value that indicates whether or not the visual element that is represented by the <code>MasterDetailPage.Master</code> property is presented to the user.
<a href="#">Master</a>	<a href="#">Page</a> . Gets or sets the master page.
<a href="#">MasterBehavior</a>	<a href="#">MasterBehavior</a> . Gets or sets a value that indicates how detail content is displayed.

# NavigationPage :

<https://developer.xamarin.com/api/type/Xamarin.Forms.NavigationPage/>

[NavigationPage\(\)](#)

Initializes a new `NavigationPage` object.

[NavigationPage\(Page\)](#)

Creates a new `NavigationPage` element with *root* as its root element.

## Public Fields

static readonly	<a href="#">BackButtonTitleProperty</a>	<code>BindableProperty</code> . Identifies the property associated with the title of the back button.
static readonly	<a href="#">BarBackgroundColorProperty</a>	<code>BindableProperty</code> . Identifies the property associated with the color of the <code>NavigationPage</code> 's bar background color.
static readonly	<a href="#">BarTextColorProperty</a>	<code>BindableProperty</code> . Identifies the property associated with the color of the <code>NavigationPage</code> 's bar text color.
static readonly	<a href="#">CurrentPageProperty</a>	<code>BindableProperty</code> . Identifies the <code>NavigationPage.CurrentPage</code> property.
static readonly	<a href="#">HasBackButtonProperty</a>	<code>BindableProperty</code> . Backing store for the <code>HasBackButton</code> property.
static readonly	<a href="#">HasNavigationBarProperty</a>	<code>BindableProperty</code> . Backing store for the <code>HasNavigationBar</code> property.
static readonly	<a href="#">TintProperty</a>	<code>BindableProperty</code> . Identifies the <code>NavigationPage.Tint</code> bindable property.
static readonly	<a href="#">TitleIconProperty</a>	<code>BindableProperty</code> . Indicates the <code>NavigationPage.SetTitleIcon/NavigationPage.GetTitleIcon</code> property.

# TabPage

- Le Xamarin.Forms TabbedPage consiste en une liste d'onglets et d'une plus grande zone d'affichage, chaque onglet chargeant son contenu dans la zone d'affichage.



# TabPage



## Public Constructors

[TabPage\(\)](#)

Creates a new `TabPage` element with default values.

## Public Fields

static readonly

[BarBackgroundColorProperty](#)

`BindableProperty`.

static readonly

[BarTextColorProperty](#)

`BindableProperty`.

## Public Properties

[BarBackgroundColor](#)

`Color`.

[BarTextColor](#)

`Color`.

## Public Methods

[On<T>\(\) : IPlatformElementConfiguration<T, TabbedPage>](#)

Returns the platform-specific instance of this `TabPage`, on which a platform-specific method may be called.

## Protected Methods

override

[CreateDefault\(Object\) : Page](#)

Creates a default page, suitable for display in this `TabPage` page, for an object.

override

[OnParentSet\(\)](#)

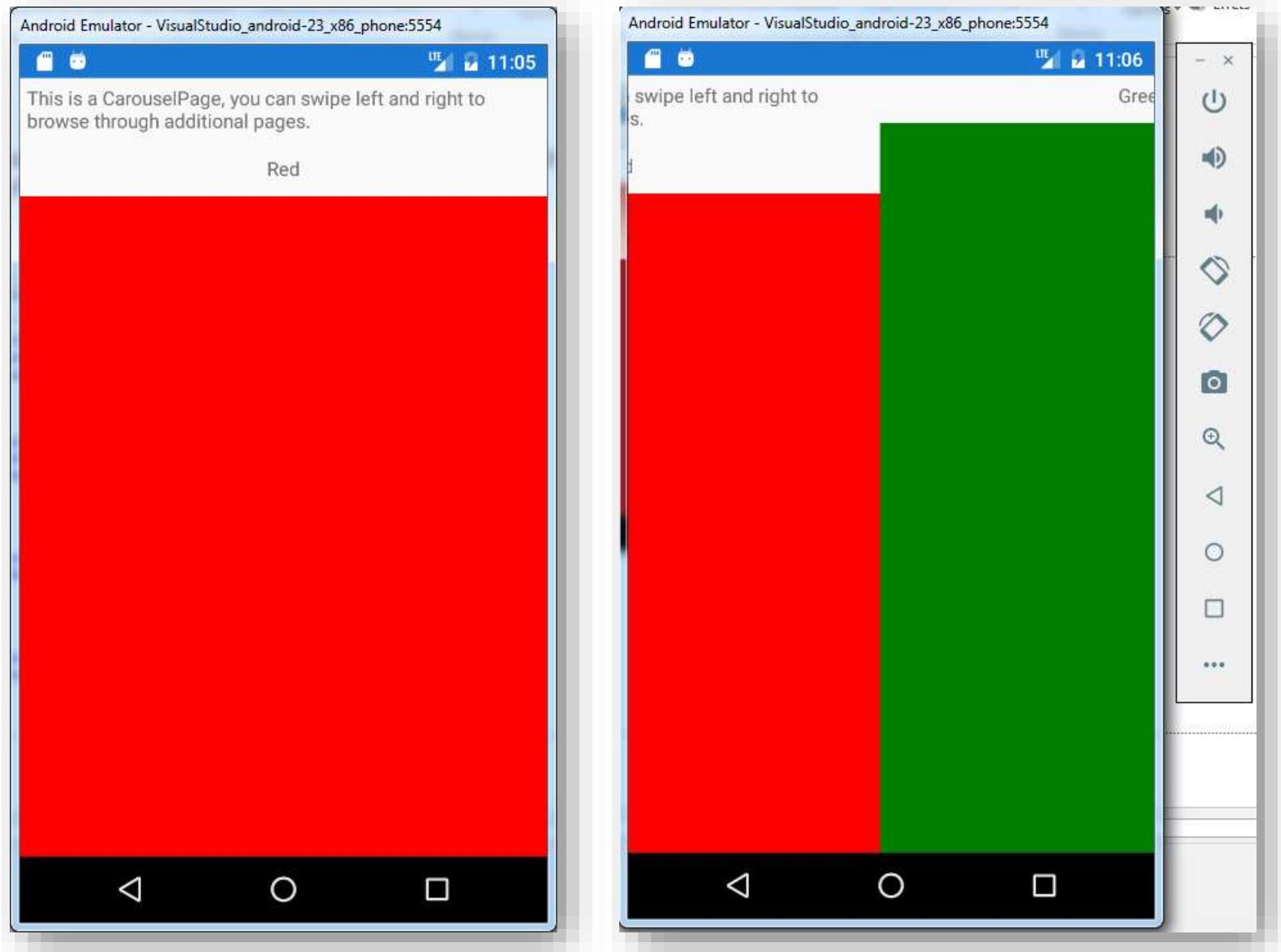
Called when the parent is set.

# TabPage: Exemple de deux pages

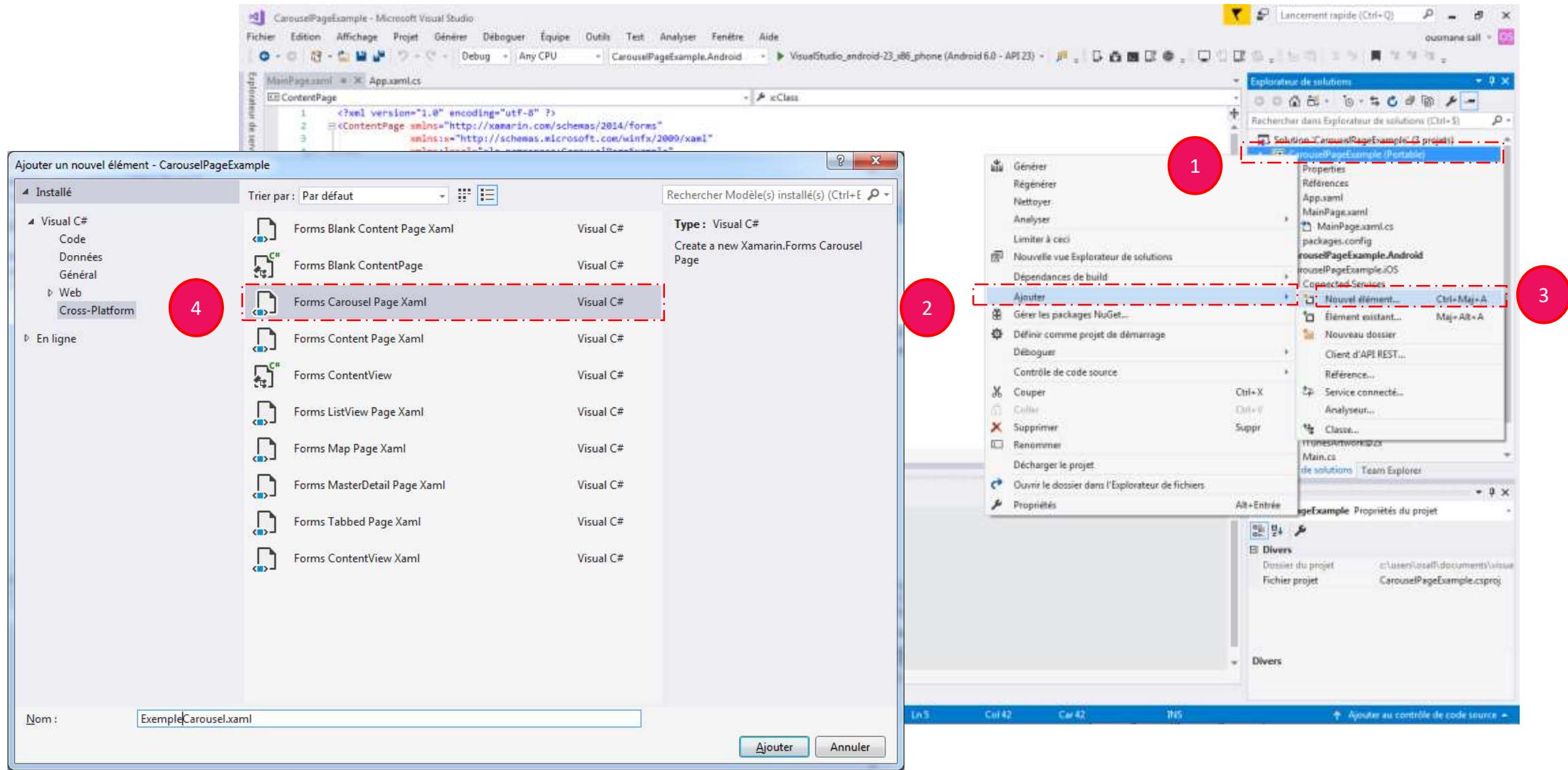
The screenshot shows the Xamarin Studio IDE interface with three code editors:

- MainPage.xaml**:  
ContentPage  
1 <?xml version="1.0" encoding="utf-8" ?>  
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"  
3 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"  
4 x:Class="ExempleTabbedPage.MainPage"  
5 Title="Page principale">  
6 <Label Text="Welcome to Xamarin Forms!"  
7 VerticalOptions="Center"  
8 HorizontalOptions="Center" />  
9 </ContentPage>
- Page1.xaml**:  
ContentPage  
1 <?xml version="1.0" encoding="utf-8" ?>  
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"  
3 xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"  
4 x:Class="ExempleTabbedPage.Page1"  
5 Title="Seconde Page">  
6 <Label Text="2éme Page"/>  
7 </ContentPage>
- App.xaml**:  
App.xaml  
1 using ...  
2  
3 namespace ExempleTabbedPage  
4 {  
5 public partial class App : Application  
6 {  
7 public App()  
8 {  
9 InitializeComponent();  
10 MainPage = new TabbedPage  
11 {  
12 Children ={  
13 new MainPage(),  
14 new Page1()  
15 }  
16 };  
17 }  
18 }  
19 }  
20 }  
21 }  
22 }  
23 }  
24 }  
25 }  
26 }  
27 }  
28 }  
29 }  
30 }  
31 }  
32 }  
33 }  
34 }  
35 }  
36 }  
37 }  
38 }  
39 }

# CarouselPage



# CarouselPage



# CarouselPage

Une page dont les utilisateurs peuvent glisser d'un côté à l'autre pour afficher des pages, comme une galerie.

- Xaml Example



The screenshot shows the XAML code for a CarouselPage example in a code editor. The file is named ExempleCarousel.xaml. The code defines a CarouselPage with three ContentPages, each containing a StackLayout with a Label and a BoxView. The labels are "Red", "Green", and "Blue". The first ContentPage also contains a descriptive text label. The StackLayouts have HorizontalOptions set to FillAndExpand, and the BoxViews have VerticalOptions set to FillAndExpand. The labels have HorizontalTextAlignment set to Center. The entire code is numbered from 1 to 25.

```
<?xml version="1.0" encoding="utf-8" ?>
<CarouselPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="CarouselPageExample.ExempleCarousel"
    Title="CarouselPage">
    <ContentPage>
        <StackLayout>
            <Label Text="This is a CarouselPage, you can swipe left and right to browse through additional pages." Margin="5" />
            <Label Text="Red" HorizontalTextAlignment="Center" HorizontalOptions="FillAndExpand" Margin="5" />
            <BoxView Color="Red" VerticalOptions="FillAndExpand" />
        </StackLayout>
    </ContentPage>
    <ContentPage>
        <StackLayout>
            <Label Text="Green" HorizontalTextAlignment="Center" HorizontalOptions="FillAndExpand" Margin="5" />
            <BoxView Color="Green" VerticalOptions="FillAndExpand" />
        </StackLayout>
    </ContentPage>
    <ContentPage>
        <StackLayout>
            <Label Text="Blue" HorizontalTextAlignment="Center" HorizontalOptions="FillAndExpand" Margin="5" />
            <BoxView Color="Blue" VerticalOptions="FillAndExpand" />
        </StackLayout>
    </ContentPage>
</CarouselPage>
```

# CarouselPage

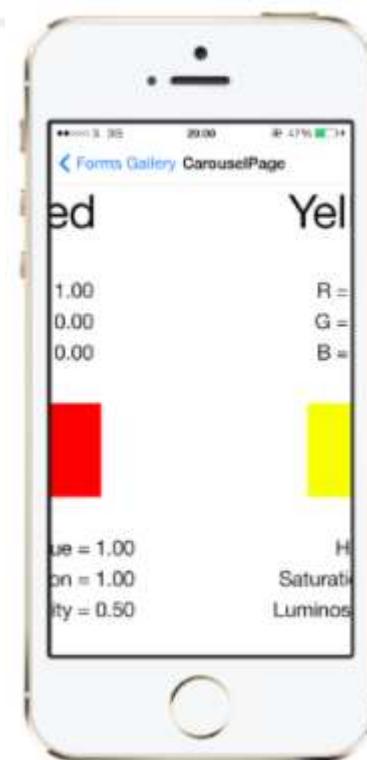
Une page dont les utilisateurs peuvent glisser d'un côté à l'autre pour afficher des pages, comme une galerie.

The following C# example creates a [CarouselPage](#) that displays three simple [ContentPage](#) elements:

## C# Example

```
List<ContentPage> pages = new List<ContentPage> (0);
Color[] colors = { Color.Red, Color.Green, Color.Blue };
foreach (Color c in colors) {
    pages.Add (new ContentPage { Content = new StackLayout {
        Children = {
            new Label { Text = c.ToString () },
            new BoxView {
                Color = c,
                VerticalOptions = LayoutOptions.FillAndExpand
            }
        }
    });
}

MainPage = new CarouselPage {
    Children = { pages [0],
        pages [1],
        pages [2] }
};
```

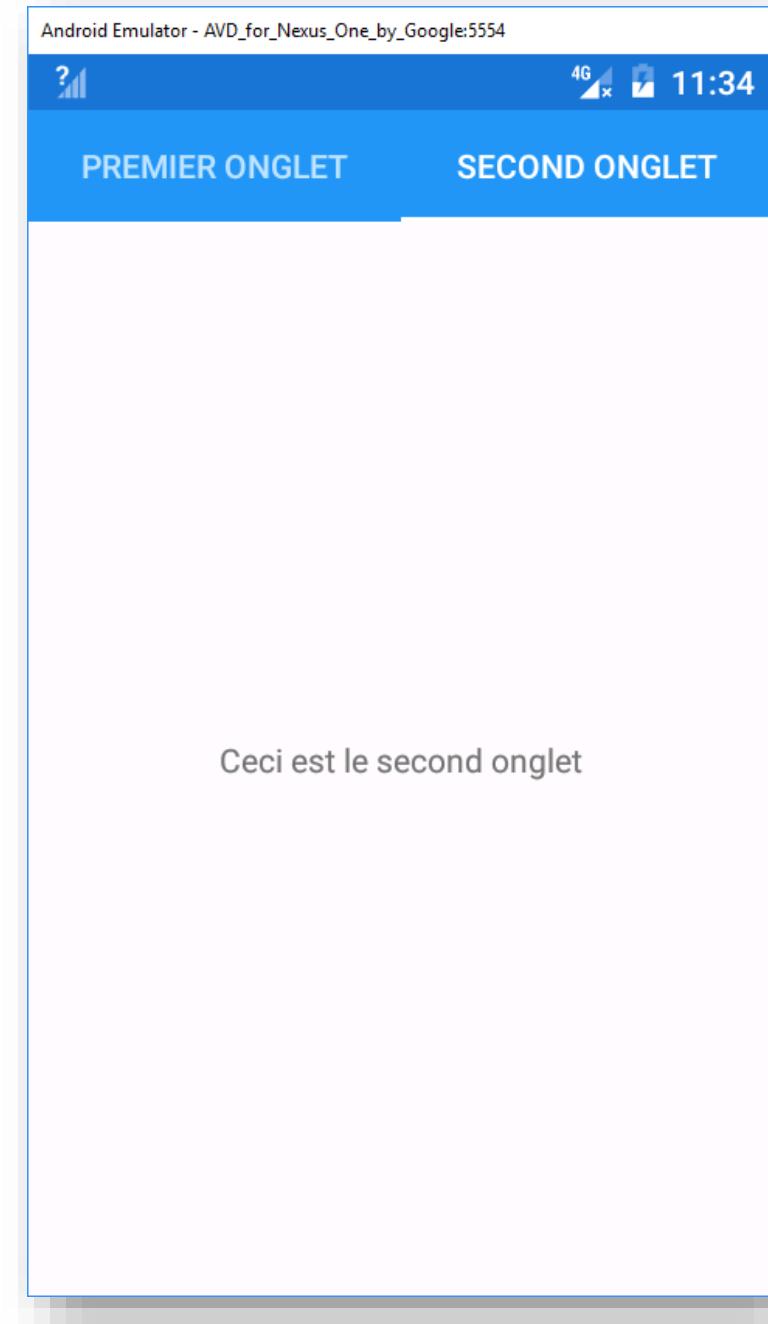


# Exercice 5

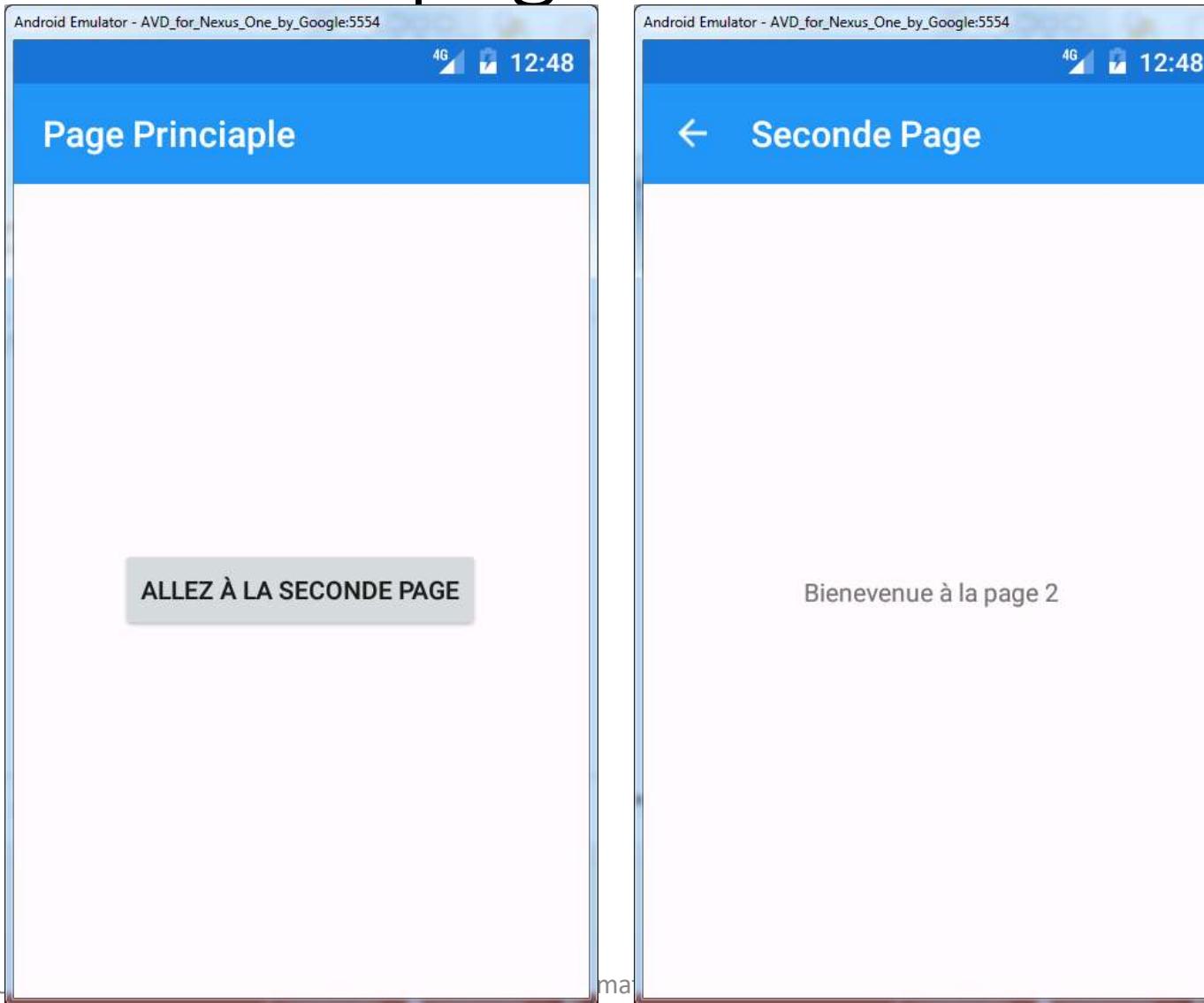
- Etudier l'exemple sur ce lien  
<https://developer.xamarin.com/api/type/Xamarin.Forms.MasterDetailPage/> et expliquer le fonctionnement dans ce contexte de MasterDetailPage

# Exercice 6

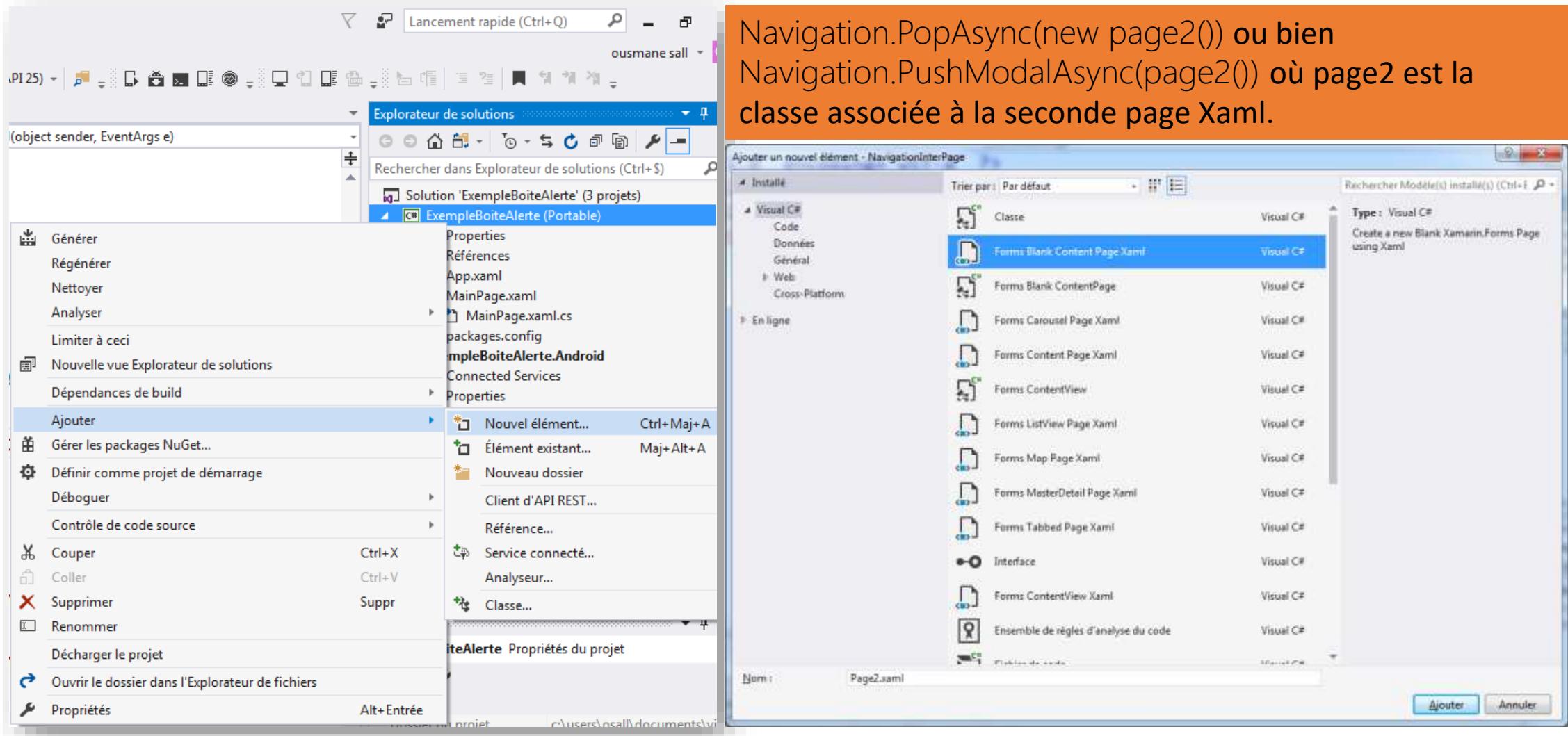
- Créez une application avec deux onglets
  - Le premier onglet contient une Label qui affiche «ceci est le premier onglet»
  - Le deuxième onglet contient une Label qui affiche «ceci est le second onglet»



# Naviguer d'une page à une autre



# Naviguer d'une page à une autre



Page2.xaml

ContentPage

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             x:Class="NavigationInterPage.Page2"
5             Title="Seconde Page">
6
7     <Label Text="Biennevenue à la page 2"
8           VerticalOptions="Center"
9           HorizontalOptions="Center"/>
10
11 </ContentPage>
```

MainPage.xaml

ContentPage

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             xmlns:local="clr-namespace:NavigationInterPage"
5             x:Class="NavigationInterPage.MainPage"
6             Title="Page Principe">
7
8     <Button Text="Allez à la seconde page"
9           VerticalOptions="Center"
10          HorizontalOptions="Center"
11          Clicked="Button_Clicked"/>
12
13 </ContentPage>
```

App.xaml

NavigationInterPage

```
1 using ...
2
3 namespace NavigationInterPage
4 {
5     public partial class App : Application
6     {
7         public App()
8         {
9             InitializeComponent();
10
11             MainPage = new NavigationPage(new MainPage());
12
13         }
14
15         protected override void OnStart()
16         {
17
18         }
19
20         protected override void OnSleep()
21         {
22
23         }
24
25         protected override void OnResume()
26         {
27
28         }
29
30     }
31
32 }
```

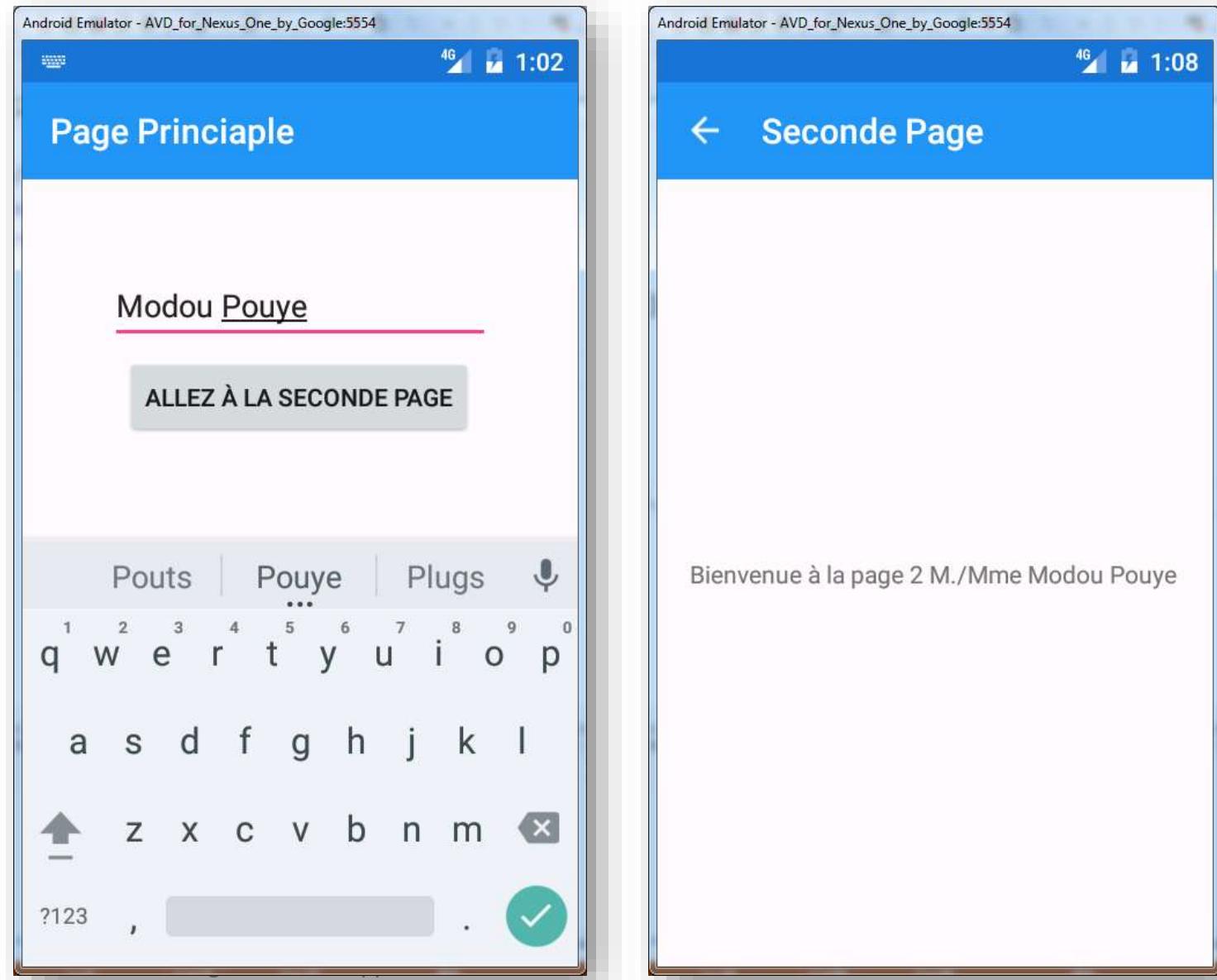
MainPage.xaml.cs

NavigationInterPage

```
1 using ...
2
3 namespace NavigationInterPage
4 {
5     public partial class MainPage : ContentPage
6     {
7         public MainPage()
8         {
9
10             private async void Button_Clicked(object sender, EventArgs e)
11             {
12                 await Navigation.PushAsync(new Page2());
13             }
14
15         }
16
17     }
18 }
```

# Naviguer d'une page à une autre: transmettre des données

- Modifions les pages afin qu'une données saisie dans la page principale puisse être transmise à la seconde



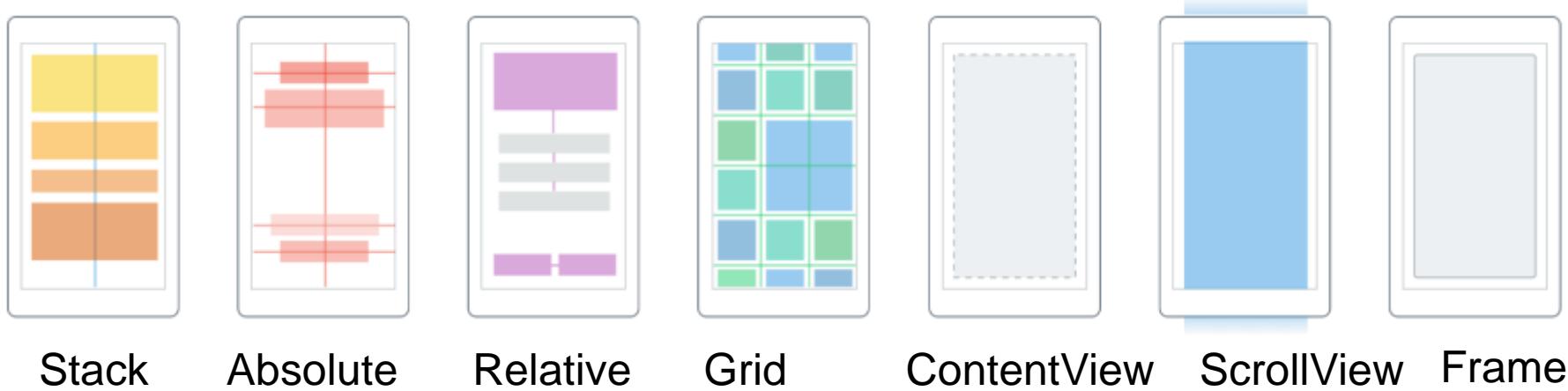
# Naviguer d'une page à une autre: transmettre des données

The image shows a Xamarin Studio interface with four code editors:

- MainPage.xaml**: XAML code for the main page. It contains a StackLayout with a text entry field (x:Name="nomSaisie") and a button (Text="Allez à la seconde page"). The button's Clicked event is bound to "Button\_Clicked".
- MainPage.xaml.cs**: C# code for the main page. It defines a MainPage class that inherits from ContentPage. It includes a constructor and a Button\_Clicked method that retrieves the value from the text entry field and performs a navigation to a new page.
- Page2.xaml**: XAML code for the second page. It contains a Label (x:Name="message") with the text "Bienvenue à la page 2".
- Page2.xaml.cs**: C# code for the second page. It defines a Page2 class that inherits from ContentPage. It has a constructor that takes a string parameter (nom) and initializes the message label with a welcome message including the name.

# Les Layouts: Types de contrôles conteneurs

# Layouts



**A l'intérieur d'une page on retrouve des Layouts :** Beaucoup d'options disponibles de quelque chose de simple comme un Stack à quelque chose de complexes et puissantes comme un Grid.

## PROPERTY      VALUE

Content      A list of [View](#) objects that represent the visual content of the [ContentPage](#). This tag can be omitted, and the contents listed directly.

# ContentPage

The screenshot shows the Microsoft Visual Studio interface for a Xamarin.Forms application named "MaPremiereAppXamarin".

**MainPage.xaml Content:**

```

1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4   xmlns:local="clr-namespace:MaPremiereAppXamarin"
5   x:Class="MaPremiereAppXamarin.MainPage">
6
7   <ContentPage.Content>
8     <AbsoluteLayout>
9       <Label Text="I'm centered on iPhone 4 but no other device"
10      AbsoluteLayout.LayoutBounds="115,150,100,100" LineBreakMode="WordWrap" />
11       <Label Text="I'm bottom center on every device."
12      AbsoluteLayout.LayoutBounds=".5,.1,.5,.1" AbsoluteLayout.LayoutFlags="All"
13      LineBreakMode="WordWrap" />
14       <BoxView Color="Olive" AbsoluteLayout.LayoutBounds="1,.5,.25,.100"
15      AbsoluteLayout.LayoutFlags="PositionProportional" />
16       <BoxView Color="Red" AbsoluteLayout.LayoutBounds=".8,.5,.25,.100"
17      AbsoluteLayout.LayoutFlags="PositionProportional" />
18       <BoxView Color="Blue" AbsoluteLayout.LayoutBounds=".5,.0,.100,.25"
19      AbsoluteLayout.LayoutFlags="PositionProportional" />
20     </AbsoluteLayout>
21   </ContentPage.Content>
22 </ContentPage>

```

**Android Emulator Preview:**

The emulator displays two labels. The top label reads "I'm centered on iPhone 4 but no other device". The bottom label reads "I'm bottom center on every device." The UI uses AbsoluteLayout and BoxView components.

**Bottom Navigation Bar:**

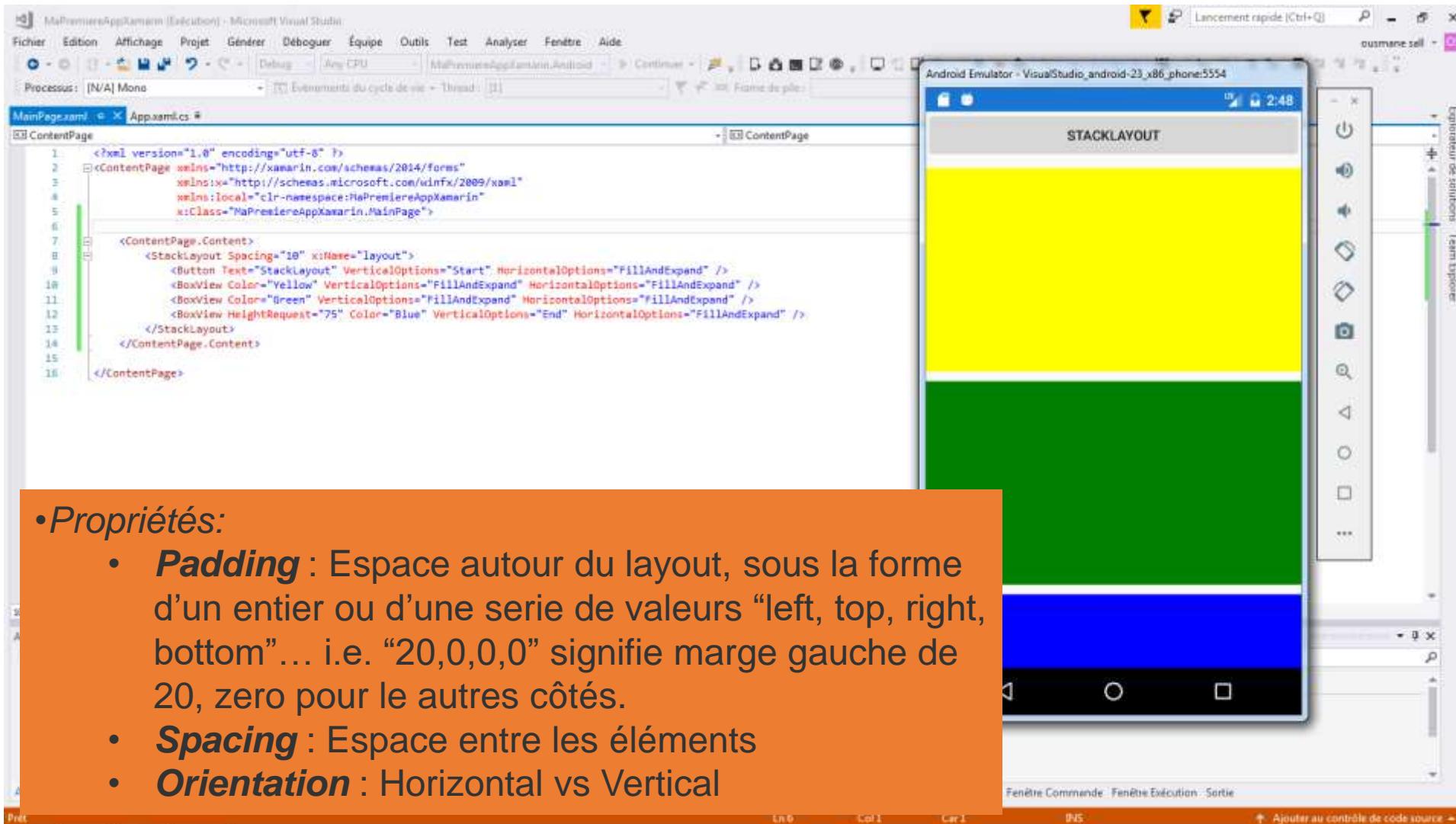
- Automatique
- Variables locales
- Espion 1
- Automatique
- Paramètres d'exception
- Arrêter en cas d'exception levé
- C++ Exceptions
- Common Language Runtime Exceptions
- GPU Memory Access Exceptions
- Java Exceptions
- JavaScript Runtime Exceptions
- Managed Debugging Assistant

**Page Footer:**

Prof. Ousmane SALL, Univ. Thiès, SN  
Programmation Applications Mobiles  
300

# StackLayout

Le layout le plus simple à utiliser  
Permet d'ajouter des éléments en horizontal ou vertical  
**Orientation="Vertical" / Orientation="Horizontal"**



The screenshot shows the Microsoft Visual Studio interface. On the left, the XAML code for a `ContentPage` is displayed:

```
1 <Xml version="1.0" encoding="utf-8" >
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4   xmlns:local="clr-namespace:MaPremiereAppXamarin"
5   x:Class="MaPremiereAppXamarin.MainPage">
6
7   <ContentPage.Content>
8     <StackLayout Spacing="10" x:Name="layout">
9       <Button Text="StackLayout" VerticalOptions="Start" HorizontalOptions="FillAndExpand" />
10      <BoxView Color="Yellow" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand" />
11      <BoxView Color="Green" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand" />
12      <BoxView HeightRequest="75" Color="Blue" VerticalOptions="End" HorizontalOptions="FillAndExpand" />
13    </StackLayout>
14  </ContentPage.Content>
15
16</ContentPage>
```

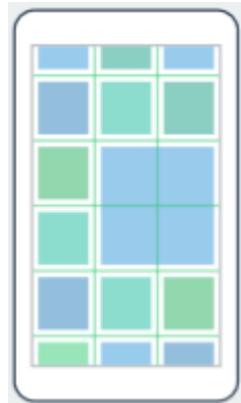
On the right, the Android Emulator window titled "Android Emulator - VisualStudio\_android-23\_x86\_phone:5554" shows the application running. The title bar says "STACKLAYOUT". The screen contains four colored boxes: a large yellow box at the top, a green box below it, a smaller blue box at the bottom, and a white space at the bottom. The Visual Studio interface also includes a Solution Explorer, a Properties window, and a Task List.

• Propriétés:

- **Padding** : Espace autour du layout, sous la forme d'un entier ou d'une série de valeurs "left, top, right, bottom" ... i.e. "20,0,0,0" signifie marge gauche de 20, zero pour les autres côtés.
- **Spacing** : Espace entre les éléments
- **Orientation** : Horizontal vs Vertical

# Grid

- La Grid vous permet de positionner vos éléments en les alignant sur une grille dont vous fixez le nombre de colonnes et de lignes.
- Auto : prend la place nécessaire
- \* : toute la place disponible
- 42 : 42 pixels de large (déconseillé)



```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="42" />
        <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>

    <Label Text="Title"
        Grid.Row="0"
        Grid.Column="0"
        Grid.ColumnSpan="2"
        />
    <StackLayout Grid.Row="1"
        Grid.Column="0"/>
    <StackLayout Grid.Row="1"
        Grid.Column="1"/>
</Grid>
```

# Grid

XAML for Xamarin.Forms supports the following properties for the [Grid](#) class:

PROPERTY	VALUE
Children	Nested visual elements that are displayed in the Grid.
ColumnDefinitions	A list of <a href="#">ColumnDefinition</a> specifications. See <a href="#">ColumnDefinition</a> .
ColumnSpacing	An integer.
RowDefinitions	A list of <a href="#">RowDefinition</a> specifications. See <a href="#">RowDefinition</a> .
RowSpacing	An integer.

XAML for Xamarin.Forms supports the following attached properties for the [Grid](#) class:

ATTACHED PROPERTY	VALUE
Column	An integer that represents the Column in which the item will appear.
ColumnSpan	An integer that represents the number of Columns that the item will span.
Row	An integer that represents the row in which the item will appear.
RowSpan	An integer that represents the number of rows that the item will span.

# Grid

```
<Grid>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="1*" />
        <ColumnDefinition Width="2*" />
    </Grid.ColumnDefinitions>
```

- Et comment je peux faire si je veux deux colonnes avec la première qui prend 1/3 de l'écran et la seconde, les 2/3 restants ?
  - La solution consiste à préfixer le symbole étoile par un nombre.
  - Dans notre cas, on mettrait la première colonne à 1\* et la deuxième colonne à 2\*.
  - L'étoile devient en fait une variable x ce qui nous donne une équation simple de la forme  $1x + 2x = \text{largeur\_écran}$
- Avantage : vous pouvez faire des interfaces qui s'adaptent à la taille de l'écran de l'utilisateur.

# Grid

In XAML:

```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="*" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="*" />
        <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>
    <Label Text="Top Left" Grid.Row="0" Grid.Column="0" />
    <Label Text="Top Right" Grid.Row="0" Grid.Column="1" />
    <Label Text="Bottom Left" Grid.Row="1" Grid.Column="0" />
    <Label Text="Bottom Right" Grid.Row="1" Grid.Column="1" />
</Grid>
```

In C#:

```
var grid = new Grid();
grid.RowDefinitions.Add(new RowDefinition { Height = new GridLength(1, GridUnitType.Star)});
grid.RowDefinitions.Add(new RowDefinition { Height = new GridLength(1, GridUnitType.Star)});
grid.ColumnDefinitions.Add(new ColumnDefinition { Width = new GridLength(1, GridUnitType.Star)});
grid.ColumnDefinitions.Add(new ColumnDefinition { Width = new GridLength(1, GridUnitType.Star)});
var topLeft = new Label { Text = "Top Left" };
var topRight = new Label { Text = "Top Right" };
var bottomLeft = new Label { Text = "Bottom Left" };
var bottomRight = new Label { Text = "Bottom Right" };
grid.Children.Add(topLeft, 0, 0);
grid.Children.Add(topRight, 0, 1);
grid.Children.Add(bottomLeft, 1, 0);
grid.Children.Add(bottomRight, 1, 1);
```



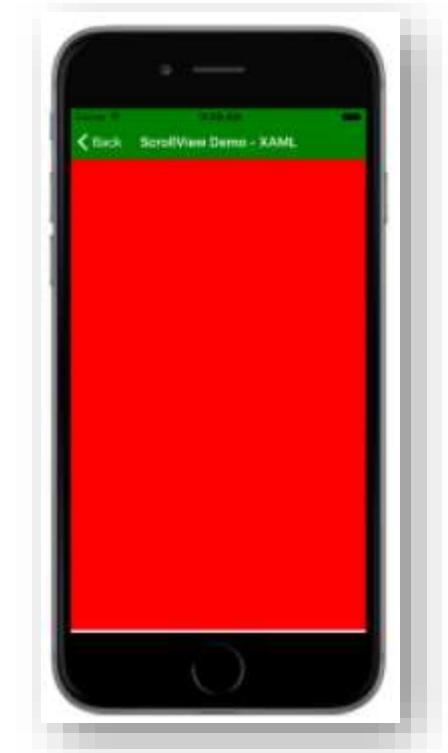
# ScrollView

- ScrollView contient les layouts et permet de defiler hors écran. ScrollView est également utilisé pour permettre aux vues de se déplacer automatiquement vers la partie visible de l'écran lorsque le clavier affiche.
- ScrollView a les propriétés suivantes :
  - **Content** – Obtient ou définit la vue à afficher dans le ScrollView.
  - **ContentSize** – read-only, obtient la taille du contenu, qui a une largeur et une hauteur.
  - **Orientation** – This is a ScrollOrientation, which is an enumeration that can be set to Horizontal, Vertical, or Both.
  - **ScrollX** – read-only, gets the current scroll position in the X dimension.
  - **ScrollY** – read-only, gets the current scroll position in the Y dimension.

# ScrollView

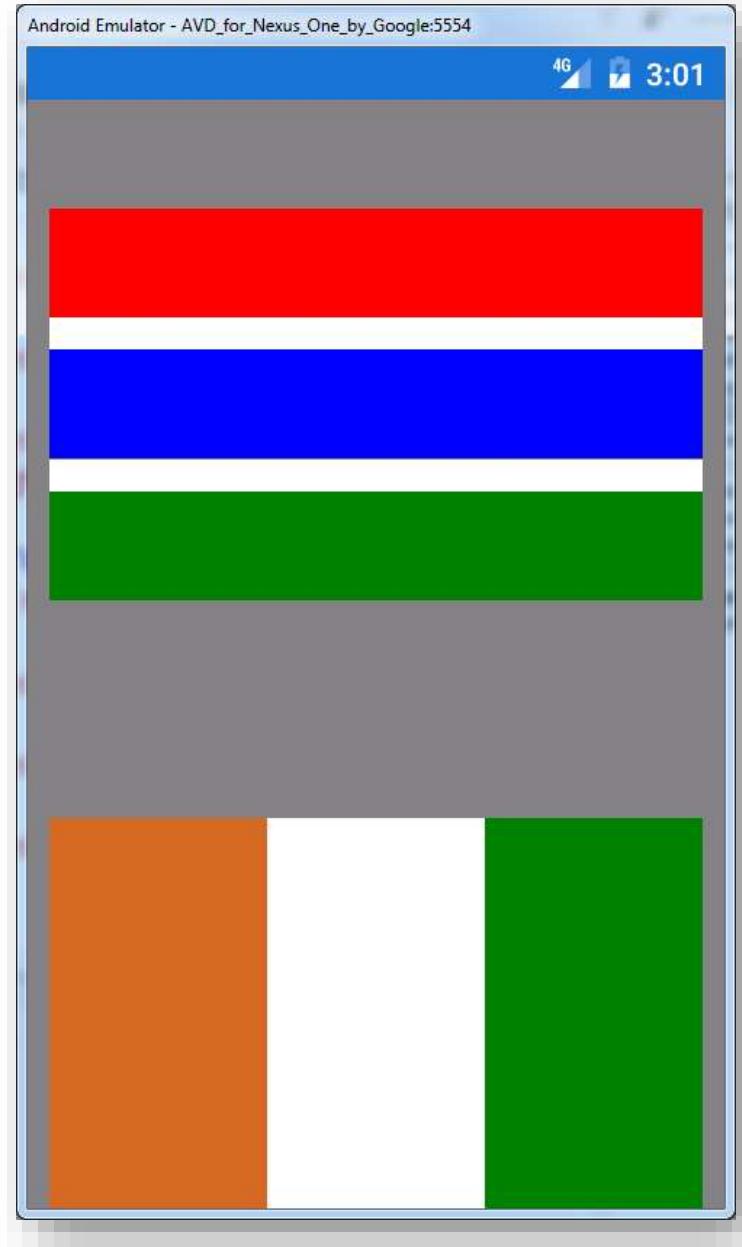
- Avant le scrolling de l'utilisateur seulement le BoxView est visible

```
<ContentPage.Content>
    <ScrollView>
        <StackLayout>
            <BoxView BackgroundColor="Red" HeightRequest="600" WidthRequest="150" />
            <Entry />
        </StackLayout>
    </ScrollView>
</ContentPage.Content>
```



# Exercice 7

- En utilisant StackLayout comme conteneur créer l'application affichant les drapeaux Gambien et Ivoirien.
- En utilisant le conteneur Grid créer l'application affichant les drapeaux malien et français.



# Avec StackLayout: une solution possible

```
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      x:Class="NavigationInterPage.Drapeaux"
5      Title="Drapeaux"
6      BackgroundColor="Gray">
7      <StackLayout Orientation="Vertical" Padding="10,50,10,50" VerticalOptions="Start" Spacing="0"
8          HeightRequest="200">
9          <BoxView Color="Red" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
10         <BoxView Color="White" HorizontalOptions="FillAndExpand" HeightRequest="15"/>
11         <BoxView Color="Blue" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
12         <BoxView Color="White" HorizontalOptions="FillAndExpand" HeightRequest="15"/>
13         <BoxView Color="Green" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
14         <BoxView HorizontalOptions="FillAndExpand" HeightRequest="50"/>
15
16         <StackLayout Orientation="Horizontal" Spacing="0"
17             VerticalOptions="CenterAndExpand" HeightRequest="200">
18             <BoxView Color="Chocolate" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
19             <BoxView Color="White" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
20             <BoxView Color="Green" HorizontalOptions="FillAndExpand" HeightRequest="50"/>
21         </StackLayout>
22     </StackLayout>
23 </ContentPage>
```

# Exercice 8

- En utilisant le conteneur Grid créer l'application affichant les drapeaux malien et français.

# Exercice 9

En utilisant le Layout Grid, créer la page de la calculatrice ci-contre.



<https://developer.xamarin.com/samples/mobile/LivePlayer/BasicCalculator/BasicCalculator.zip>



# Les vues

# Views

Les vues sont des contrôles, les éléments visibles et interactifs sur une page. Celles-ci vont des **vues de base** comme des **boutons, des étiquettes et des zones de texte** aux vues plus avancées comme **les listes et la navigation**. Les vues contiennent des **propriétés** qui déterminent leur **contenu, la police, la couleur et l'alignement**. L'alignement horizontal et vertical est fixé par propriétés **HorizontalOptions** et **VerticalOptions**. Comme mise en page, les vues peuvent être rembourrées avec l'espace, dimensionné pour étendre, pour remplir l'espace disponible, ou diminuer en fonction de leur contenu.



# Les vues

- Xamarin.Forms utilise le mot View pour se référer à des objets visuels tels que des boutons, des étiquettes ou des zones de saisie de texte - ce qui peut être plus communément appelé contrôle des widgets.
- Ces éléments d'UI sont généralement des sous-classes de View.

## Protected Constructors

[View\(\)](#)

Initializes a new instance of the View class.

## Public Fields

static readonly

[HorizontalOptionsProperty](#)

[BindableProperty](#). Identifies the HorizontalOptions bindable property.

static readonly

[MarginProperty](#)

[BindableProperty](#). Backing store for the View.Margin property.

static readonly

[VerticalOptionsProperty](#)

[BindableProperty](#). Identifies the VerticalOptions bindable property.

## Public Properties

[read-only]

[GestureRecognizers](#)

IList<IGestureRecognizer>. The collection of gesture recognizers associated with this view.

[HorizontalOptions](#)

LayoutOptions. Gets or sets the LayoutOptions that define how the element gets laid in a layout cycle. This is a bindable property.

[Margin](#)

Thickness. Gets or sets the margin for the view.

[VerticalOptions](#)

LayoutOptions. Gets or sets the LayoutOptions that define how the element gets laid in a layout cycle. This is a bindable property.

<https://developer.xamarin.com/api/type/Xamarin.Forms.View/>

# Labels pour Text

- Créez une nouvelle solution PCL **Xamarin.Forms**, appelée **Salutations**, en utilisant le même processus décrit ci-dessus pour créer la solution **MaPremiereAppXamarin**. Cette nouvelle solution sera structurée plus comme un programme **Xamarin.Forms** typique, ce qui signifie qu'il définira une nouvelle classe qui découle de **ContentPage**. La plupart du temps dans ce cours, chaque classe et structure définie par un programme aura ses propres fichier. Cela signifie qu'un nouveau fichier doit être ajouté au projet Salutations:

Dans Visual Studio, vous pouvez cliquer avec le bouton droit de la souris sur le projet **Salutations** dans l'**Explorateur de solutions** et sélectionner **Ajouter > Nouvel élément** dans le menu.

À gauche de la boîte de dialogue Ajouter un nouvel élément, sélectionnez **Visual C #** et **CrossPlatform**, et dans la zone centrale, sélectionnez **Forms ContentPage**. (Attention: Il existe également une option **Forms ContentView**. Ne choisissez pas celui-là!)

# Label: Public Properties

- **XAlign, YAlign** : Propriétés d'alignement du Texte...valeurs comme **ContentAlignment.Start, TextAlignement.Center, TextAlignement.End**
- **TextColor** : Couleur du texte et couleur de fond...
- **Font** : Attributs de la famille de font, comme **Bold, Italic, Large, Medium, Small, Micro,**
- **LineBreakMode** : liée au retour à la ligne du texte du label. Valeurs possibles **CharacterWrap, NoWrap, WordWrap, HeadTruncation, MiddleTruncation, TailTruncation**

<https://developer.xamarin.com/api/type/Xamarin.Forms.Label/>

# Labels pour

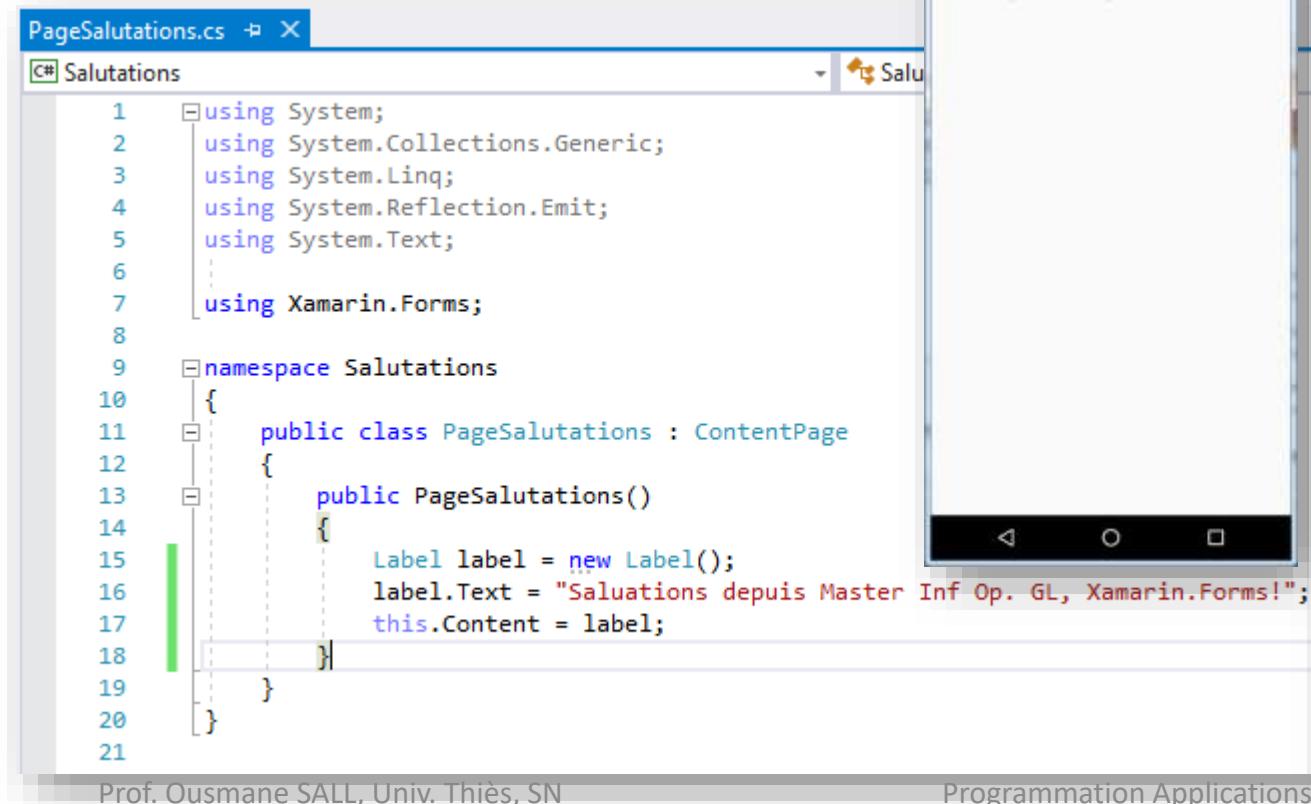
The screenshot shows the Microsoft Visual Studio interface with the following details:

- Solution Explorer:** Shows the project structure:
  - Solution 'Salutations' (1 projet)
  - Salutations (Portable)
    - Properties
    - Références
    - App.xaml
    - MainPage.xaml
    - packages.config
  - Salutations.Android
    - Properties
    - Références
- Contextual Menu:** Opened over the Salutations.Android node, showing options like Ajouter (Add), Gérer les packages NuGet..., Définir comme projet de démarrage (Set as startup project), etc.
- Code Editor:** Displays the Salutations.cs file content:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 
6 using Xamarin.Forms;
7 
8 namespace Salutations
```
- Add New Item Dialog:** A modal window titled "Ajouter un nouvel élément - Salutations".
  - Category:** Visual C# (Cross-Platform)
  - Items:**
    - Forms Blank Content Page Xaml
    - Forms Blank ContentPage** (selected)
    - Forms Carousel Page Xaml
    - Forms Content Page Xaml
    - Forms ContentView
    - Forms ListView Page Xaml
    - Forms Map Page Xaml
    - Forms MasterDetail Page Xaml
    - Forms Tabbed Page Xaml
    - Forms ContentView Xaml
  - Type:** Visual C#
  - Description:** Create a new Xamarin.Forms ContentPage with C#.
- Bottom Bar:** Contains fields for "Nom:" (Name) set to "PageSalutations.cs", and buttons "Ajouter" (Add) and "Annuler" (Cancel).
- Status Bar:** Shows "Lancement rapide (Ctrl+Q)" and "Ousmane Sall" with a user icon.
- Page-Footer:** "Prof. Ousmane SALL, Univ. Thiès, SN" and "Programmation Applications Mobiles".
- Page-Footer:** "317"

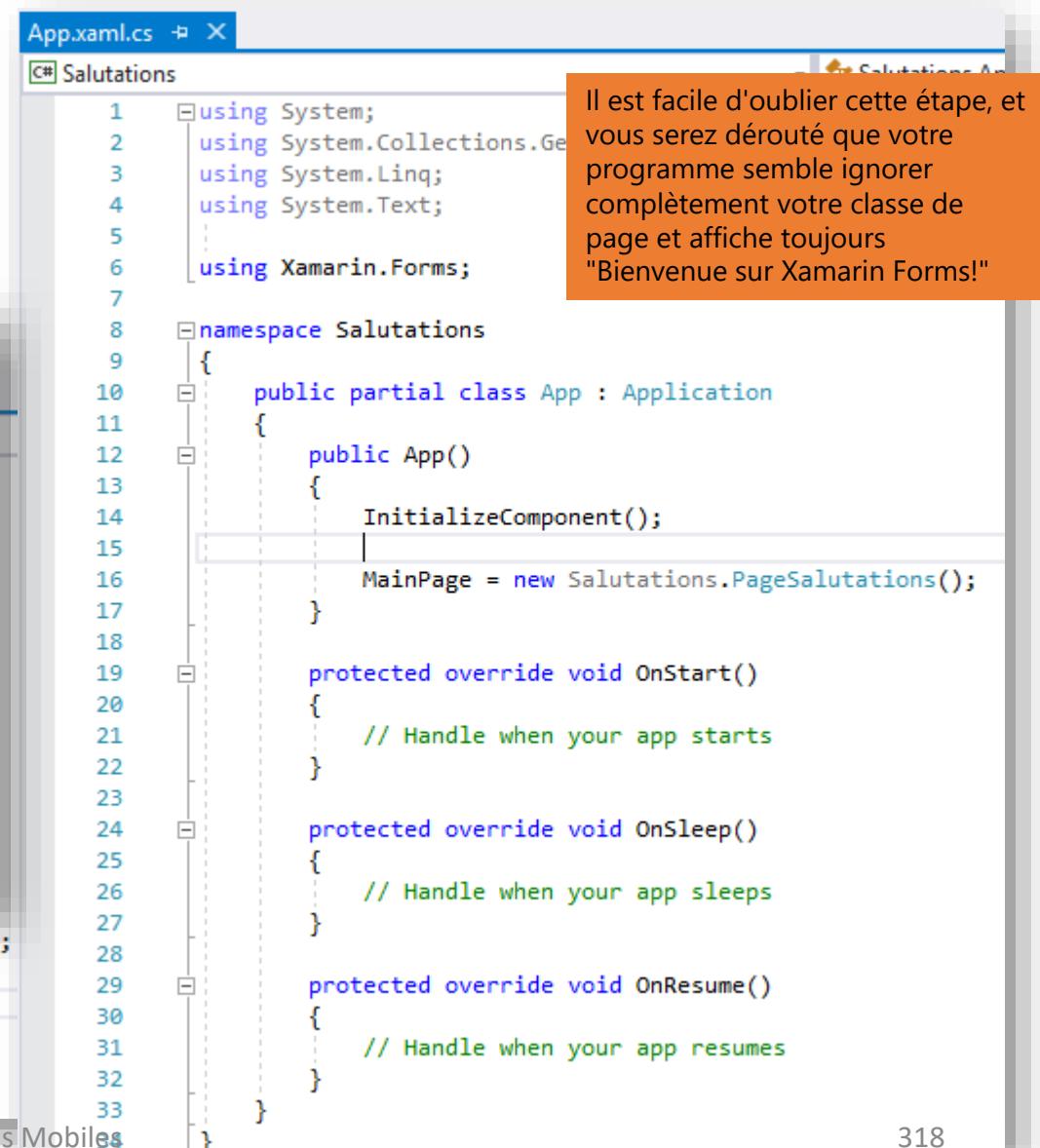
# Labels pour Text

Dans le constructeur de la classe PageSalutations, instantier un Label, définissez sa propriété de texte et définissez cette occurrence d'étiquette dans la propriété Content que PageSalutations hérite de ContentPage::



```
PageSalutations.cs  # X
C# Salutations
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Reflection.Emit;
5  using System.Text;
6
7  using Xamarin.Forms;
8
9  namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17             this.Content = label;
18         }
19     }
20 }
21
```

The screenshot shows the Visual Studio IDE with two tabs open: PageSalutations.cs and App.xaml.cs. The PageSalutations.cs tab contains C# code defining a PageSalutations class that inherits from ContentPage. Inside the constructor, it creates a new Label, sets its text to "Salutations depuis Master Inf Op. GL, Xamarin.Forms!", and sets it as the Content of the page. The App.xaml.cs tab shows the main application structure, including the App class which sets the MainPage to a new instance of PageSalutations. To the right, the Android Emulator window displays the resulting application with the specified text.



```
App.xaml.cs  # X
C# Salutations
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5
6  using Xamarin.Forms;
7
8  namespace Salutations
9  {
10     public partial class App : Application
11     {
12         public App()
13         {
14             InitializeComponent();
15             MainPage = new Salutations.PageSalutations();
16         }
17
18         protected override void OnStart()
19         {
20             // Handle when your app starts
21         }
22
23         protected override void OnSleep()
24         {
25             // Handle when your app sleeps
26         }
27
28         protected override void OnResume()
29         {
30             // Handle when your app resumes
31         }
32     }
33 }
```

A callout box on the right side of the screen contains the following note:

Il est facile d'oublier cette étape, et vous serez dérouté que votre programme semble ignorer complètement votre classe de page et affiche toujours "Bienvenue sur Xamarin Forms!"

# Inclure le padding sur la page

The screenshot shows the Visual Studio IDE. On the left, the code editor displays `PageSalutations.cs` with the following C# code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Reflection.Emit;
5  using System.Text;
6
7  using Xamarin.Forms;
8
9  namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17             this.Content = label;
18             Padding = new Thickness(0, 100, 0, 0);
19         }
20     }
21 }
22
```

On the right, the Android Emulator window titled "Android Emulator - VisualStudio\_android-x86\_phone:5554" shows the application running. The screen displays the text "Salutations depuis Master Inf Op. GL, Xamarin.Forms!" centered on the screen. The emulator interface includes a toolbar with icons for power, volume, and navigation, and a status bar at the top.

La classe de page définit une propriété nommée **Padding** qui marque une zone autour du périmètre intérieur de la page dans laquelle le contenu ne peut pas s'immiscer. La propriété **Padding** est du type **Thickness**, une structure qui définit quatre propriétés nommées Left, Top, Right, Bottom.

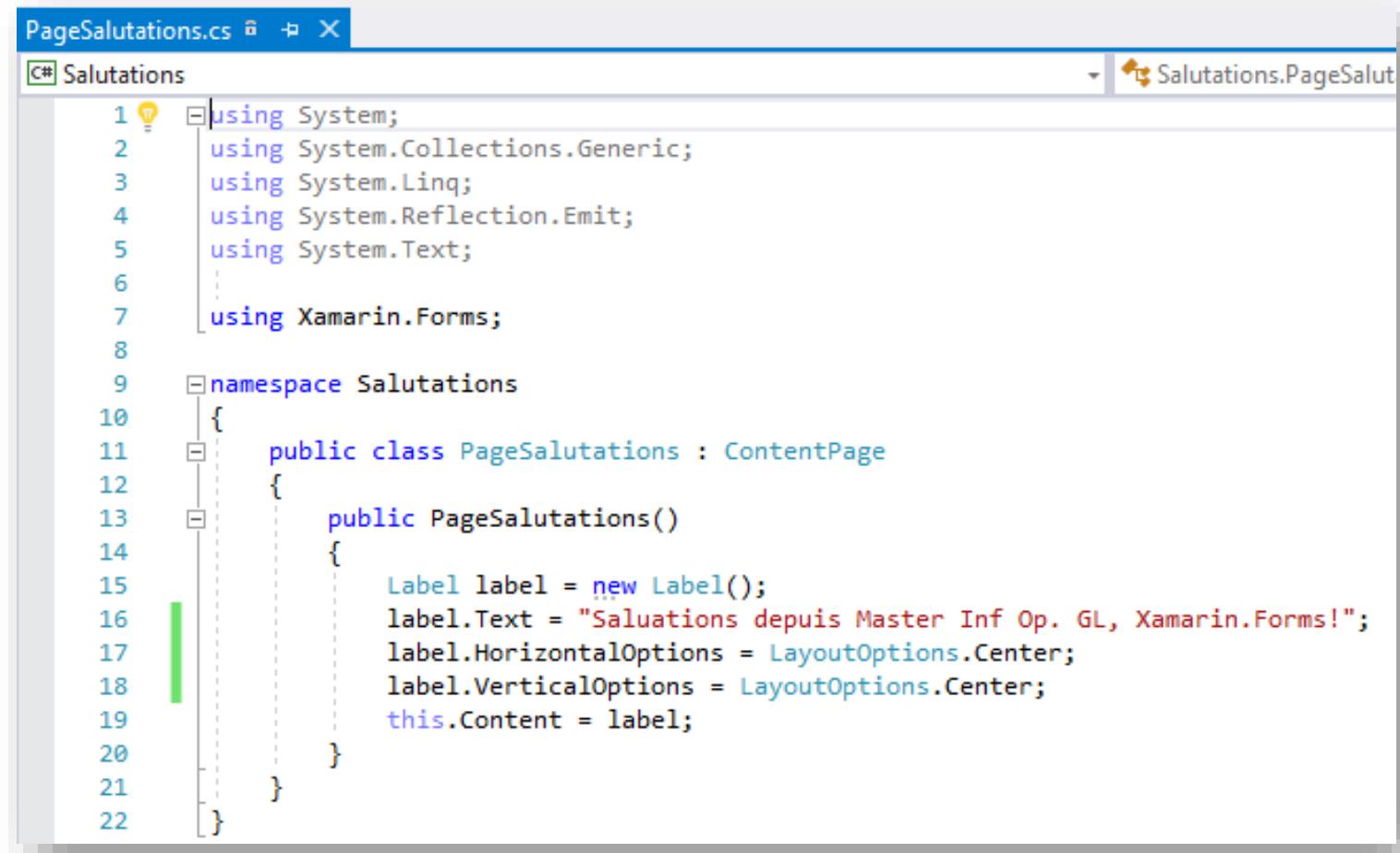
# Centrer le Label dans la page

- **Xamarin.Forms** propose un certain nombre de facilités pour la mise en page sans nécessiter que le programme effectue des calculs impliquant des tailles et des coordonnées.
- La classe **View** définit deux propriétés nommées: "**HorizontalOptions**" et "**VerticalOptions**", qui précisent comment une vue doit être positionnée par rapport à son parent (dans ce cas, **ContentPage**).
- Ces deux propriétés sont du type **LayoutOptions**, une structure exceptionnellement importante dans **Xamarin.Forms**.

# Centrer le Label dans la page

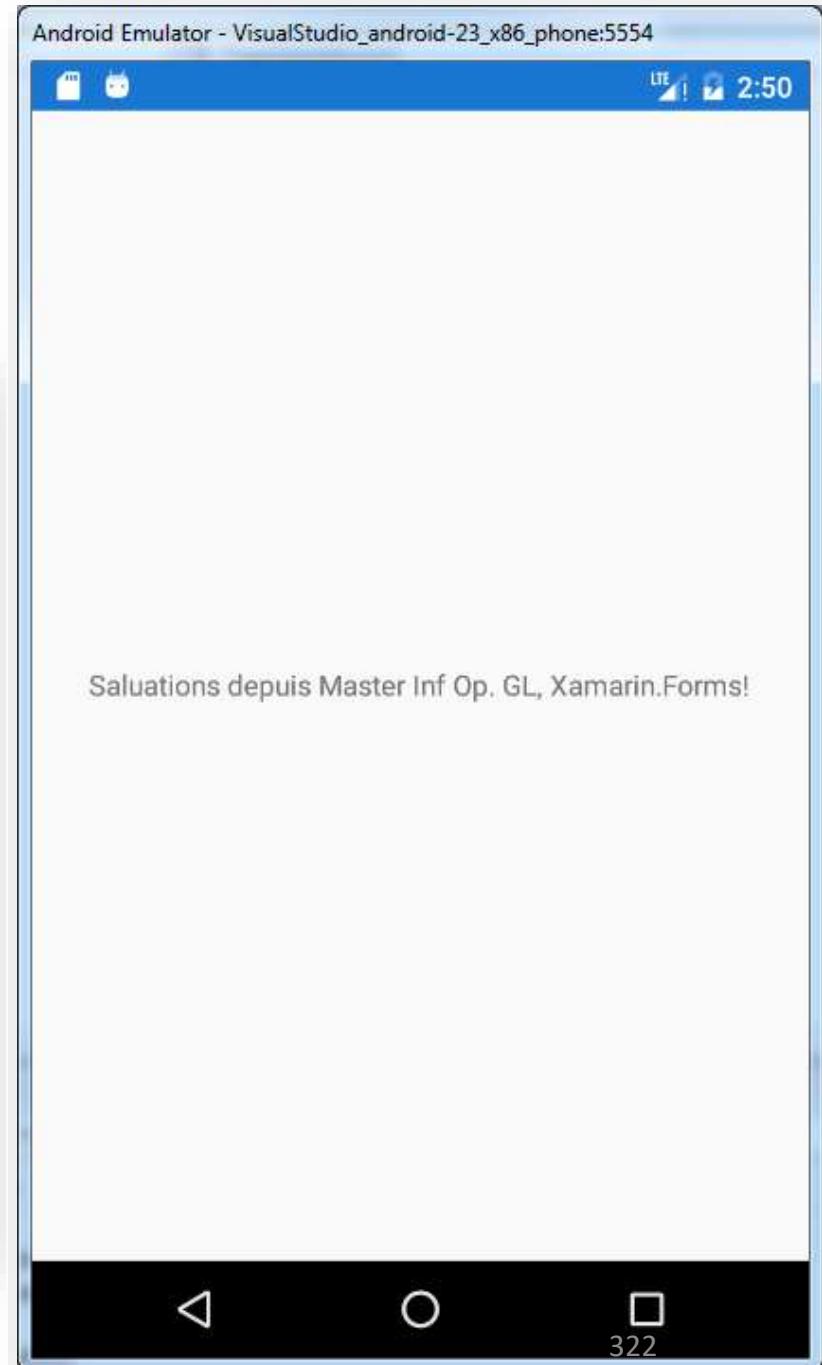
- Généralement, vous utiliserez la structure **LayoutOptions** en spécifiant l'un des huit champs statiques publics en lecture seule qui définit les valeurs retournées par **LayoutOptions** :
  - *Start*
  - *Center*
  - *End*
  - *Fill*
  - *StartAndExpand*
  - *CenterAndExpand*
  - *EndAndExpand*
  - *FillAndExpand*

# Centrer le Label dans la page



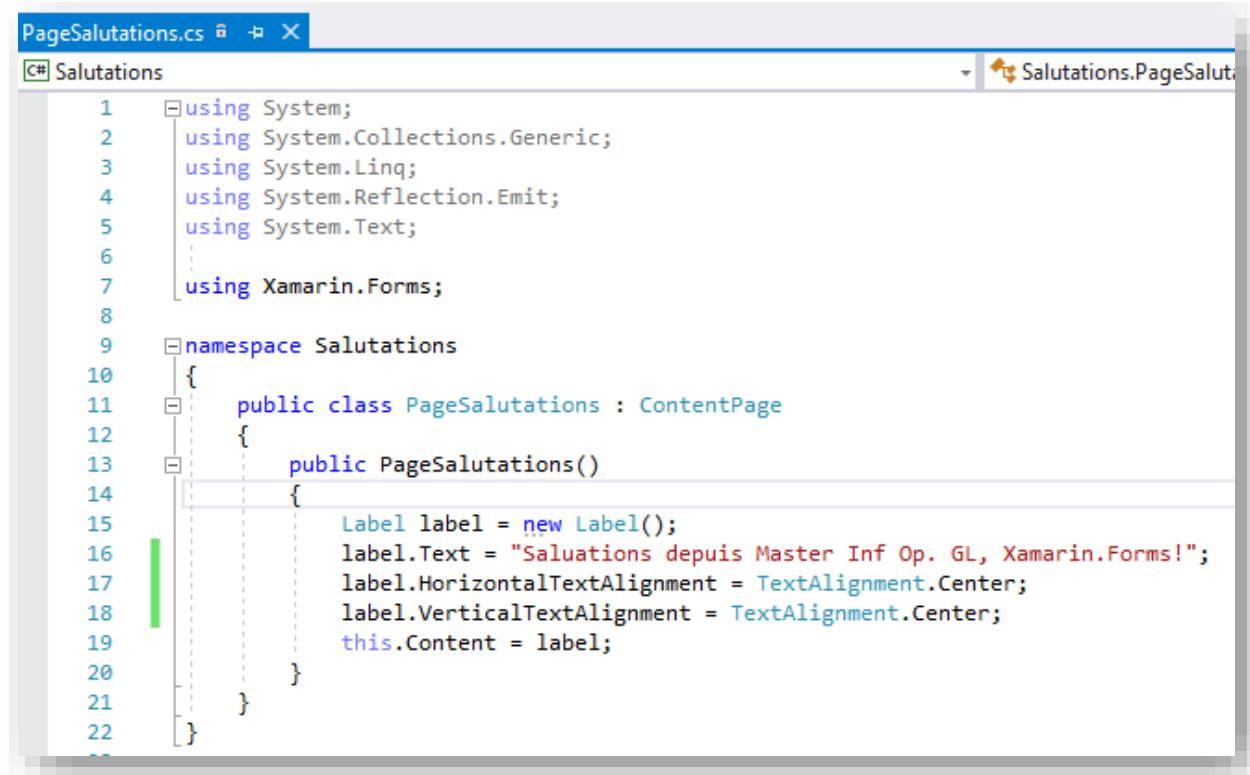
The screenshot shows the code editor for a C# file named `PageSalutations.cs`. The code defines a `PageSalutations` class that inherits from `ContentPage`. Inside the constructor, a `Label` is created and its `Text` is set to "Salutations depuis Master Inf Op. GL, Xamarin.Forms!". The `HorizontalOptions` and `VerticalOptions` properties are both set to `LayoutOptions.Center`, which centers the label both horizontally and vertically within the page's content area.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Reflection.Emit;
5 using System.Text;
6 
7 using Xamarin.Forms;
8 
9 namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17             label.HorizontalOptions = LayoutOptions.Center;
18             label.VerticalOptions = LayoutOptions.Center;
19             this.Content = label;
20         }
21     }
22 }
```

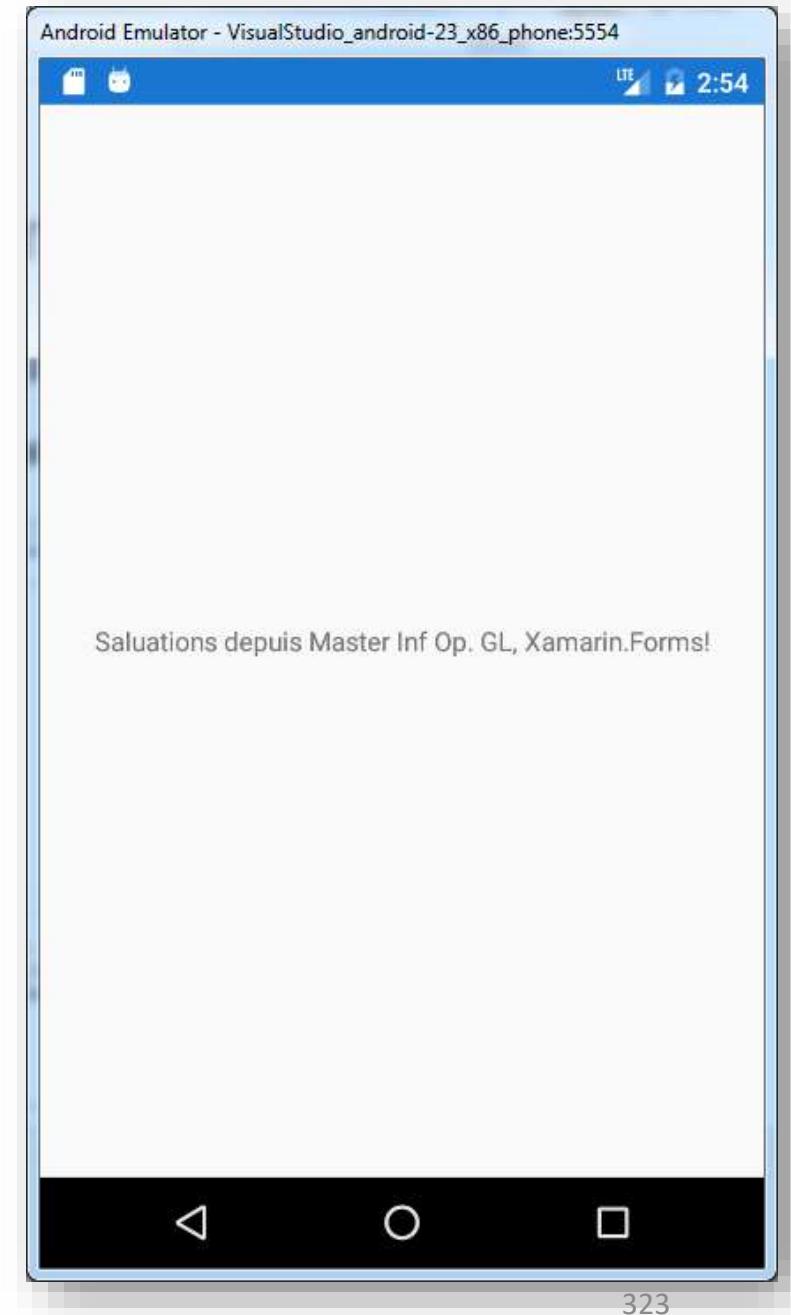


# Centrer le Label dans la page

Le Label est destiné à afficher du texte. Il est souvent souhaitable de contrôler comment les lignes de texte sont alignées horizontalement: left justified, right justified, or centered.

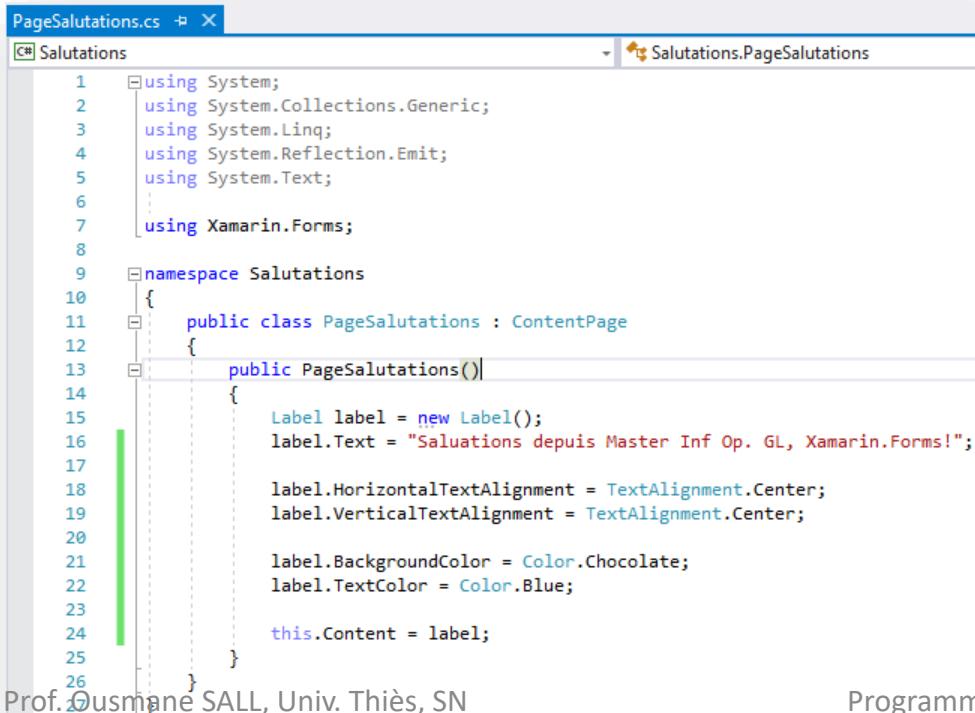


```
PageSalutations.cs  X
C# Salutations
Salutations.PageSalutat...
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Reflection.Emit;
5  using System.Text;
6
7  using Xamarin.Forms;
8
9  namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17             label.HorizontalTextAlignment = TextAlignment.Center;
18             label.VerticalTextAlignment = TextAlignment.Center;
19             this.Content = label;
20         }
21     }
22 }
```

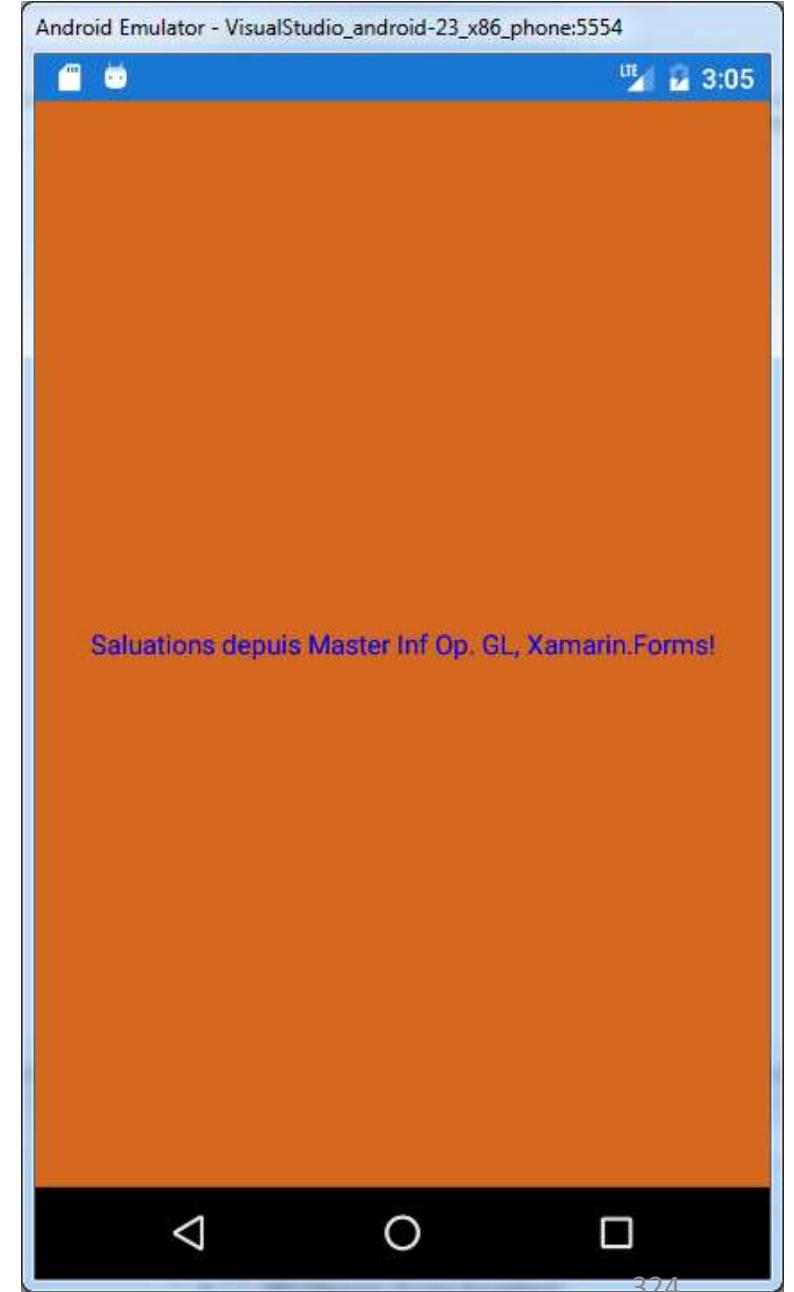


# Texte et couleur de fond

Comme vous l'avez vu, la vue Label affiche du texte dans une couleur appropriée pour l'appareil. Vous pouvez remplacer ce comportement en définissant deux propriétés, nommées `TextColor` et `BackgroundColor`. Label lui-même définit `TextColor`, mais il hérite `BackgroundColor` de `VisualElement`, ce qui signifie que la page et la mise en page ont également une propriété `BackgroundColor`.



```
PageSalutations.cs  X
C#| Salutations
Salutations.PageSalutations
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Reflection.Emit;
5  using System.Text;
6
7  using Xamarin.Forms;
8
9  namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17
18             label.HorizontalTextAlignment = TextAlignment.Center;
19             label.VerticalTextAlignment = TextAlignment.Center;
20
21             label.BackgroundColor = Color.Chocolate;
22             label.TextColor = Color.Blue;
23
24             this.Content = label;
25         }
26     }
}
```



# Texte et couleur de fond

The screenshot shows the development environment for a Xamarin.Forms application. On the left, the code editor displays the file `PageSalutations.cs` with the following content:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Reflection.Emit;
5  using System.Text;
6
7  using Xamarin.Forms;
8
9  namespace Salutations
10 {
11     public class PageSalutations : ContentPage
12     {
13         public PageSalutations()
14         {
15             Label label = new Label();
16             label.Text = "Salutations depuis Master Inf Op. GL, Xamarin.Forms!";
17
18             label.HorizontalOptions = LayoutOptions.Center;
19             label.VerticalOptions = LayoutOptions.Center;
20
21             label.BackgroundColor = Color.Chocolate;
22             label.TextColor = Color.Blue;
23
24             this.Content = label;
25         }
26     }
27 }
28
29
```

The code defines a `PageSalutations` class that inherits from `ContentPage`. It contains a single `Label` control with the text "Salutations depuis Master Inf Op. GL, Xamarin.Forms!". The label's background color is set to `Color.Chocolate` and its text color to `Color.Blue`.

On the right, the Android Emulator window titled "Android Emulator - VisualStudio\_android-23\_x86\_phone:5554" shows the resulting application. The label is centered on the screen with the specified blue text and chocolate-colored background. The status bar at the top of the emulator shows the time as 3:09.

# Font sizes et attributs

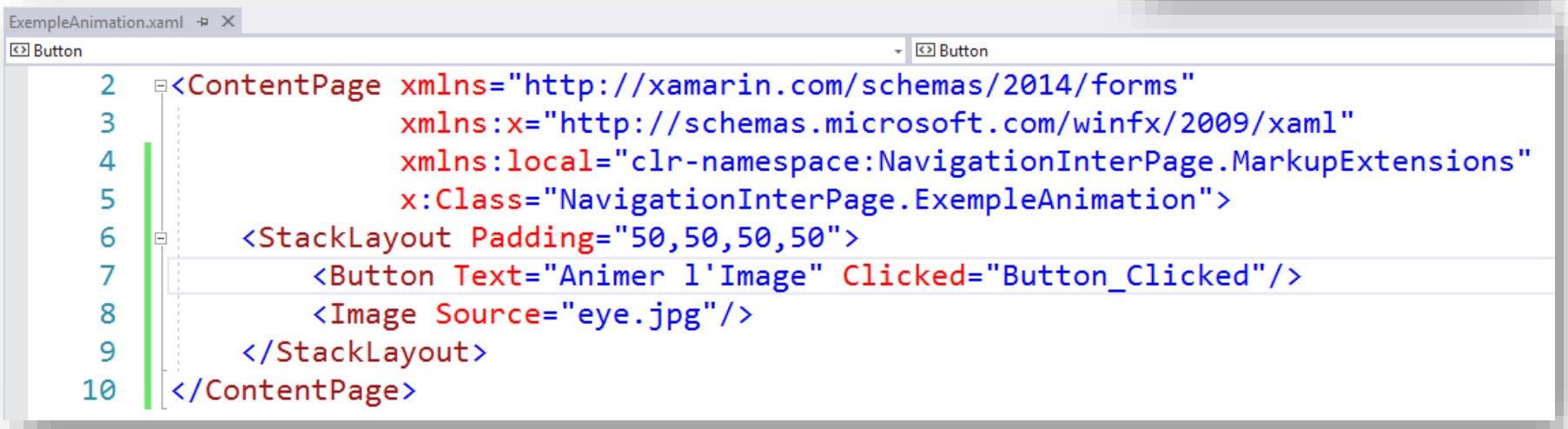
- Par défaut, Label utilise une police système définie par chaque plate-forme, mais Label définit également plusieurs propriétés que vous pouvez utiliser pour modifier cette police. L'étiquette est l'une des deux classes seulement avec ces propriétés liées à la police; Button est l'autre. Les propriétés qui vous permettent de modifier cette police sont :
  - `FontFamily` de type `string`
  - `FontSize` de type `double`
  - `FontAttributes` de type `FontAttributes`, une enumeration de trois membres: `None`, `Bold` et `Italic`.

# Image

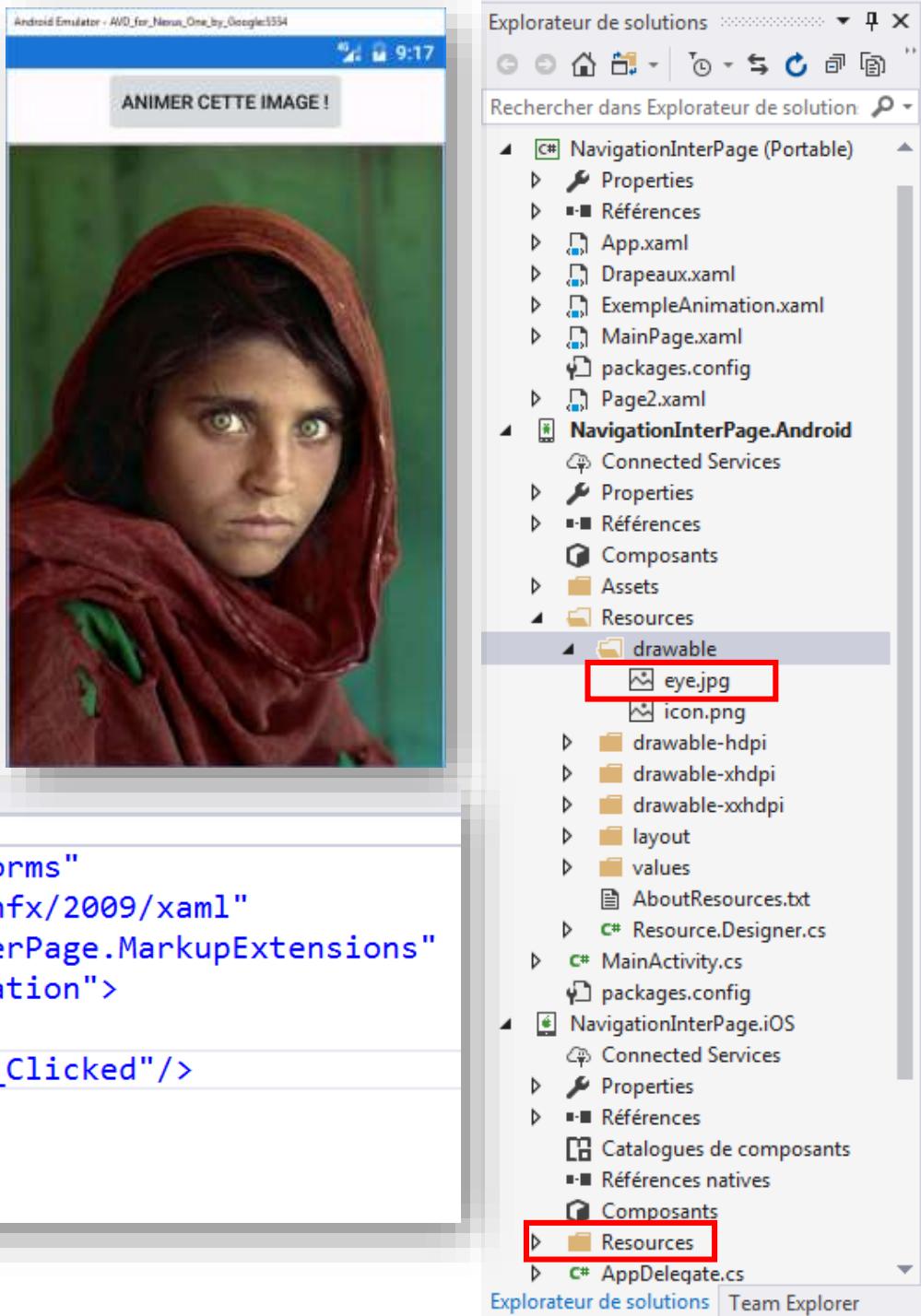
- Affiche une image.
- Les Attributs:
  - *Aspect* : Scaling of image...stretch or fill. Values include **Fill**, **AspectFill**, **AspectFit**
  - *IsEnabled*, *IsFocused*, *IsLoading*, *IsOpaque*, *IsVisible* : Drive behaviors of image control
  - *Source*: This is where the image is sourced from...can be local resource or online.

# Image

- Copier et coller votre image dans **Resources/Drawable du projet android** et pour IOS dans le répertoire **Resources**

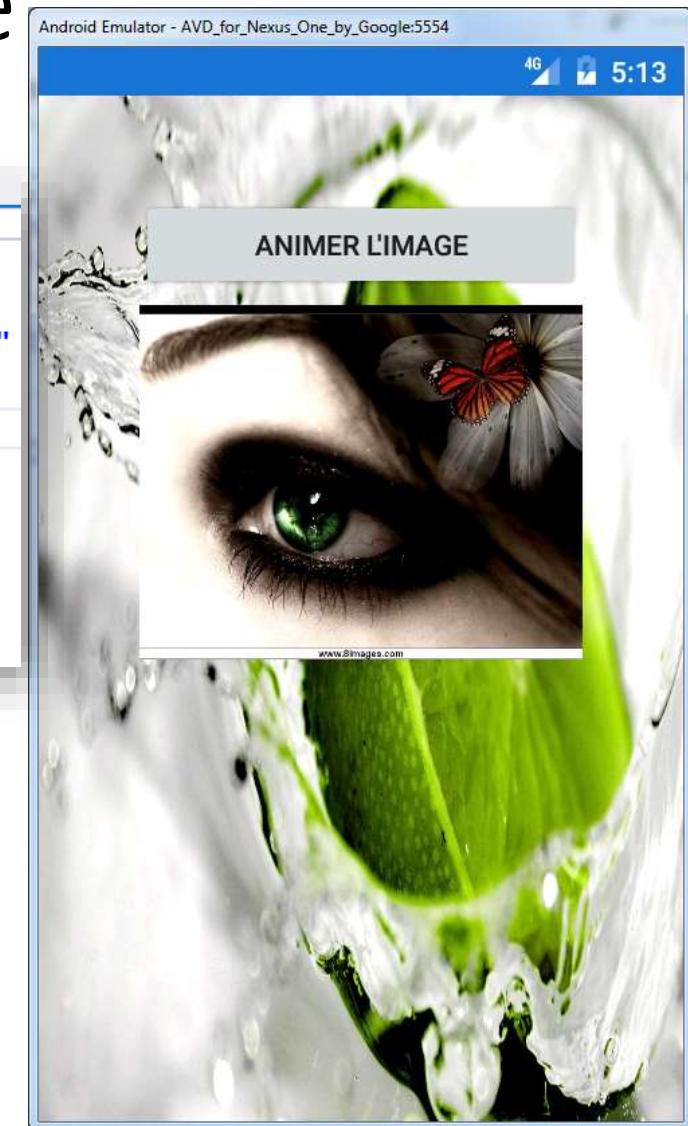


```
ExempleAnimation.xaml
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:local="clr-namespace:NavigationInterPage.MarkupExtensions"
    x:Class="NavigationInterPage.ExempleAnimation">
    <StackLayout Padding="50,50,50,50">
        <Button Text="Animer l'Image" Clicked="Button_Clicked"/>
        <Image Source="eye.jpg"/>
    </StackLayout>
</ContentPage>
```



# Image : Image de fond pour la Page

```
ExempleAnimation.xaml  X
ContentPage
2 <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             xmlns:local="clr-namespace:NavigationInterPage.MarkupExtensions"
5             x:Class="NavigationInterPage.ExempleAnimation"
6             BackgroundImage="lime_splash.jpg">
7     <StackLayout Padding="50,50,50,50">
8         <Button Text="Animer l'Image" Clicked="Button_Clicked"/>
9         <Image Source="eye.jpg"/>
10    </StackLayout>
11 </ContentPage>
```



# Aminations

- Xamarin.Forms comprend sa propre infrastructure d'animation qui est simple pour créer des animations simples, tout en étant suffisamment polyvalent pour créer des animations complexes.
- Les classes d'animation Xamarin.Forms ciblent différentes propriétés des éléments visuels, avec une animation typique modifiant progressivement une propriété d'une valeur à l'autre sur une période de temps. Notez qu'il n'y a pas d'interface XAML pour classes d'animation Xamarin.Forms. Cependant, les animations peuvent être encapsulées dans des comportements puis référencées à partir de XAML.

# Animations Simples : Rotating, scaling, translating, et fading

- La classe **ViewExtensions** fournit des méthodes d'extension qui peuvent être utilisées pour construire des animations simples:
  - Traduire pour animer les propriétés **TranslationX** et **TranslationY** d'un **VisualElement**.
  - **ScaleTo** anime la propriété **Scale** d'un **VisualElement**.
  - **RelScaleTo** applique une augmentation incrémentielle animée ou une diminution à la propriété **Scale** d'un **VisualElement**.
  - **RotateTo** anime la propriété **Rotation** d'un **VisualElement**.
  - **RelRotateTo** applique une augmentation ou une diminution incrémentielle animée à la propriété **Rotation** d'un **VisualElement**.
  - **RotateXTo** anime la propriété **RotationX** d'un **VisualElement**.
  - **RotateYTo** anime la propriété **RotationY** d'un **VisualElement**.
  - **FadeTo** anime la propriété **Opacity** d'un **VisualElement**.
- Par défaut, chaque animation dure 250 millisecondes. Cependant, une durée pour chaque animation peut être spécifiée lors de la création de l'animation.

# Single Animations: rotation

- L'exemple de code suivant illustre l'utilisation de la méthode **RotateTo** pour animer la propriété Rotation d'une image:

Ce code anime l'instance d'Image en tournant jusqu'à 360 degrés en 2 secondes (2000 millisecondes). La méthode **RotateTo** obtient la valeur de la propriété Rotation actuelle pour le début de l'animation, puis tourne de cette valeur à son premier argument (360). Une fois l'animation terminée, la propriété Rotation de l'image est réinitialisée à 0. Cela garantit que la propriété Rotation ne reste pas à 360 après la fin de l'animation, ce qui empêcherait des rotations supplémentaires.

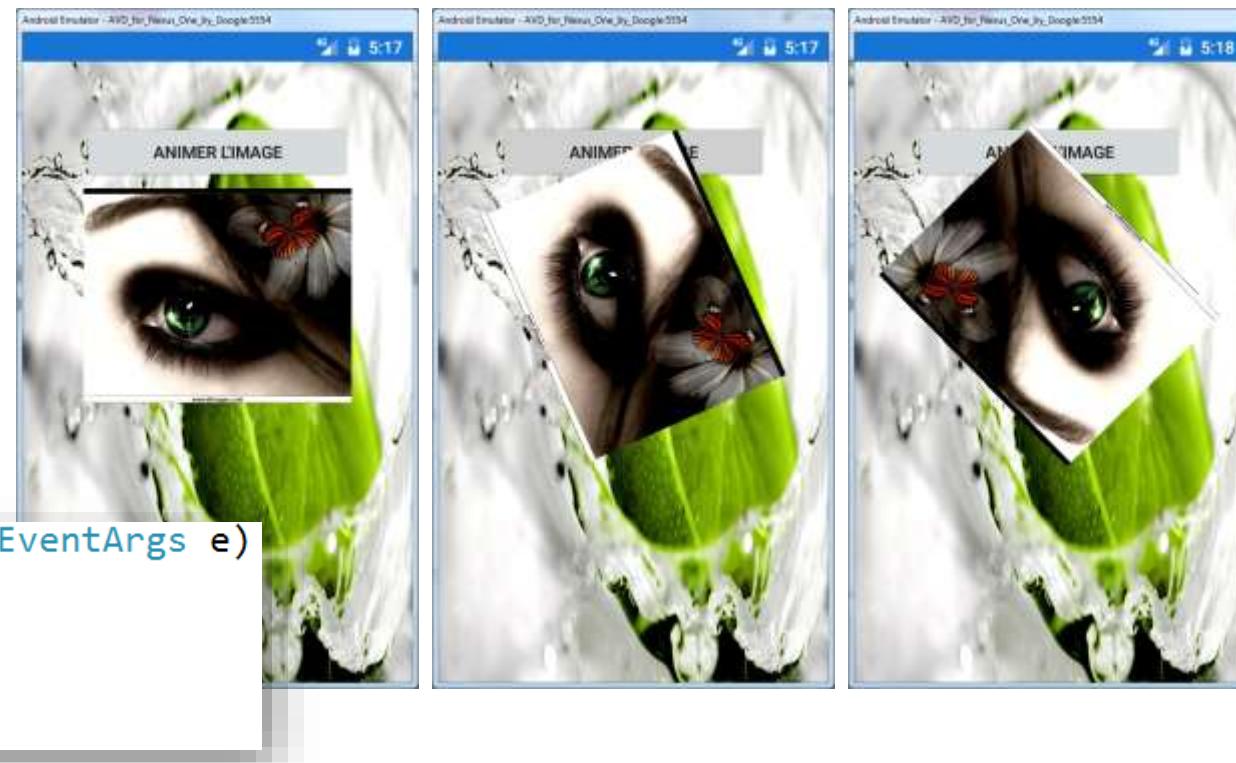
Les captures d'écran suivantes montrent la rotation en cours sur chaque plate-forme:

```
private async void Button_Clicked(object sender, EventArgs e)
{
    await monImage.RotateTo(360, 2000);
    monImage.Rotation = 0;
}
```



```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:local="clr-namespace:NavigationInterPage.MarkupExtensions"
    x:Class="NavigationInterPage.ExempleAnimation">
    <StackLayout Padding="50,50,50,50">
        <Button Text="Animer l'Image" Clicked="Button_Clicked"/>
        <Image x:Name="monImage" Source="eye.jpg"/>
    </StackLayout>
</ContentPage>
```

```
await image.RotateTo (360, 2000);
image.Rotation = 0;
```



# Single Animations: relative rotation

- L'exemple de code suivant illustre l'utilisation de la méthode **RelRotateTo** pour augmenter ou diminuer progressivement la propriété Rotation d'une image: **await image.RelRotateTo (360, 2000);**
- Ce code anime l'instance d'image en tournant à 360 degrés depuis sa position de départ sur 2 secondes (2000 millisecondes).



# Animations simples: mise à l'échelle(Scaling))

- L'exemple de code suivant illustre l'utilisation de la méthode **ScaleTo** pour animer la propriété Scale d'une image: attendez image.ScaleTo (3, 5000);
- Ce code anime l'instance Image en augmentant jusqu'à trois fois sa taille sur 5 secondes (5000 millisecondes). La méthode **ScaleTo** obtient la valeur actuelle de la valeur de la valeur (valeur par défaut de 1) pour le début de l'animation, puis passe de cette valeur à son premier argument (2). Cela a pour effet d'élargir la taille de l'image à deux fois sa taille.



```
private async void Button_Clicked(object sender, EventArgs e)
{
    await monImage.ScaleTo(3, 5000);
    await monImage.ScaleTo(1,0);
}
```

# Animations simples: traduction

- L'exemple de code suivant illustre l'utilisation de la méthode **TranslateTo** pour animer les propriétés TranslationX et TranslationY d'une image: attendez `image.TranslateTo (-100, -100, 1000);`
- Ce code anime l'instance Image en la traduisant horizontalement et verticalement sur 1 seconde (1000 millisecondes). La méthode TranslateTo translate simultanément l'image 100 pixels vers la gauche et 100 pixels vers le haut. C'est parce que les premier et deuxième arguments sont à la fois des nombres négatifs. Fournir des nombres positifs translatant l'image vers la droite et vers le bas.

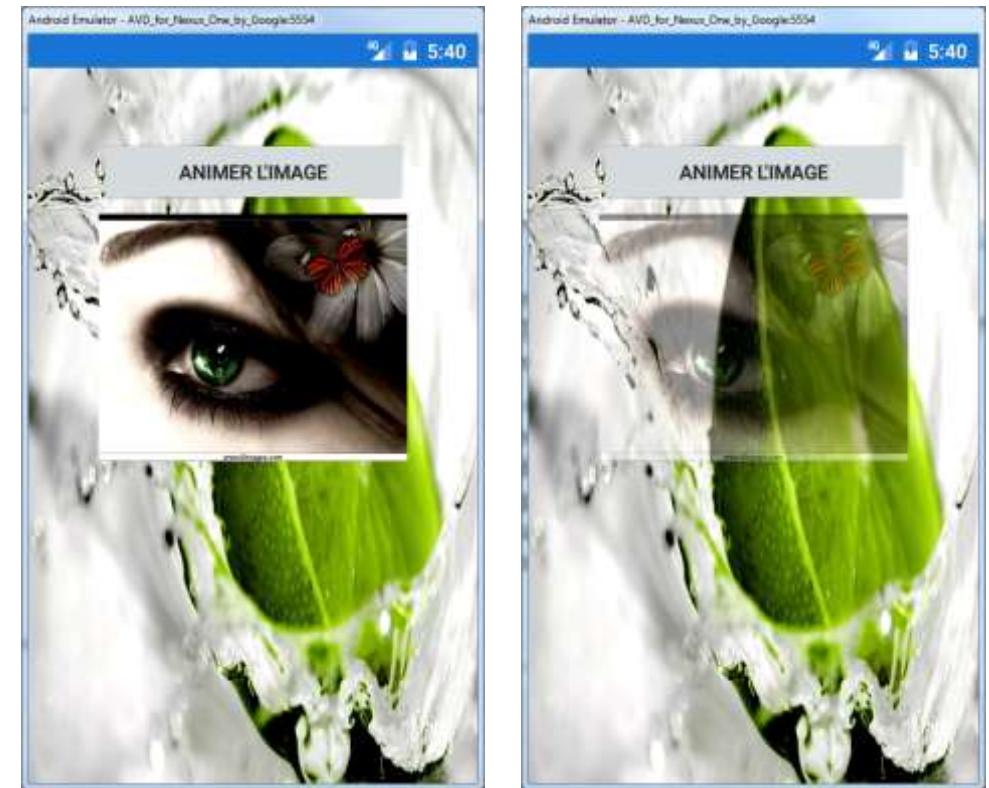
```
private async void Button_Clicked(object sender, EventArgs e)
{
    await monImage.TranslateTo(-100, -100, 1000);
    await monImage.TranslateTo(0, 0, 0);
}
```



# Single Animations: Fading

- L'exemple de code suivant illustre l'utilisation de la méthode **FadeTo** pour animer la propriété **Opacity** d'une image:
- Ce code anime l'instance Image en la décolorant en plus pendant 5 secondes (5000 millisecondes). La méthode **FadeTo** obtient la valeur de propriété Opacity actuelle pour le début de l'animation, puis s'efface de cette valeur à son premier argument (1).

```
private async void Button_Clicked(object sender, EventArgs e)
{
    monImage.Opacity = 0;
    await monImage.FadeTo(1, 5000);
}
```



# Couleurs

- Xamarin.Forms offre une classe de couleurs multiplate-forme flexible.
- La classe Couleur fournit un certain nombre de méthodes pour créer une instance de couleur:
  - Couleurs nommées - une collection de couleurs nommées communes, y compris le rouge, le vert et le bleu.
  - FromHex - valeur de chaîne similaire à la syntaxe utilisée dans HTML, par exemple "00FF00". Alpha est facultativement spécifié comme première paire de caractères ("CC00FF00").
  - FromHsla - Hue, saturation et luminosité double valeur, avec valeur alpha optionnelle (0.0-1.0).
  - FromRgb - Rouge, vert et bleu int valeurs (0-255).
  - FromRgba - Valeurs rouge, vert, bleu et alpha int (0-255).
  - FromUint - définissez une seule double valeur représentant argb.

# Couleurs

```
var red    = new Label { Text = "Red",   BackgroundColor = Color.Red };
var orange = new Label { Text = "Orange",BackgroundColor = Color.FromHex("FF6A00") };
var yellow = new Label { Text = "Yellow",BackgroundColor = Color.FromHsla(0.167, 1.0, 0.5, 1.0) };
var green  = new Label { Text = "Green",  BackgroundColor = Color.FromRgb (38, 127, 0) };
var blue   = new Label { Text = "Blue",   BackgroundColor = Color.FromRgba(0, 38, 255, 255) };
var indigo = new Label { Text = "Indigo",BackgroundColor = Color.FromRgb (0, 72, 255) };
var violet = new Label { Text = "Violet",BackgroundColor = Color.FromHsla(0.82, 1, 0.25, 1) };

var transparent = new Label { Text = "Transparent",BackgroundColor = Color.Transparent };
var @default = new Label { Text = "Default",   BackgroundColor = Color.Default };
var accent = new Label { Text = "Accent",     BackgroundColor = Color.Accent };
```



# Couleurs: Using from XAML

- Les couleurs peuvent également être facilement référencées dans XAML en utilisant les noms de couleurs définis ou les représentations hexadécimales affichées ici :

- `<Label Text="Sea color" BackgroundColor="Aqua" />`
- `<Label Text="RGB" BackgroundColor="#00FF00" />`
- `<Label Text="Alpha plus RGB" BackgroundColor="#CC00FF00" />`
- `<Label Text="Tiny RGB" BackgroundColor="#0F0" />`
- `<Label Text="Tiny Alpha plus RGB" BackgroundColor="#C0F0" />`

# ActivityIndicator

- Un contrôle visuel utilisé pour indiquer que quelque chose est en cours. Ce contrôle donne un indice visuel à l'utilisateur que quelque chose se passe, sans information sur ses progrès.
- Un indicateur qui montre qu'il y a un traitement d'action et que l'utilisateur doit attendre qu'il soit complété. Propriétés communes :
  - ***Color*** : Couleur de l'indicateur
  - ***IsEnabled, IsFocused, IsRunning IsVisible*** : Valeurs pour conduire les comportements des indicateurs d'activité

# BoxView

- Une vue utilisée pour dessiner un rectangle de couleur solide. BoxView est un support utile pour les images ou les éléments personnalisés lors du prototypage initial. BoxView a taille par défaut de 40x40. Si vous avez besoin d'une taille différente, affectez des valeurs à `VisualElement.WidthRequest` et `VisualElement.HeightRequest`.
- Similar to Rectangle in Windows XAML. Used to display a box with some color on a page. Some properties:
  - ***Color*** : Color of box
  - ***Opacity*** : Value of opacity...
  - ***IsEnabled*, *IsFocused*** : properties of Box



## XAML Example

```
<BoxView Color="Red" />
```

The following example shows a basic use:

## C# Example

```
using System;
using Xamarin.Forms;

namespace FormsGallery
{
    class BoxViewDemoPage : ContentPage
    {
        public BoxViewDemoPage()
        {
            Label header = new Label
            {
                Text = "BoxView",
                Font = Font.BoldSystemFontOfSize(50),
                HorizontalOptions = LayoutOptions.Center
            };

            BoxView boxView = new BoxView
            {
                Color = Color.Accent,
                WidthRequest = 150,
                HeightRequest = 150,
                HorizontalOptions = LayoutOptions.Center,
                VerticalOptions = LayoutOptions.CenterAndExpand
            };

            // Accommodate iPhone status bar.
            this.Padding = new Thickness(10, Device.OnPlatform(20, 0, 0), 10, 5);

            // Build the page.
            this.Content = new StackLayout
            {
                Children =
                {
                    header,
                    boxView
                }
            };
        }
    }
}
```



# Button

- Used to trigger event processing in response to user's actions.

Common properties:

- **BorderColor, BorderRadius, BorderWidth** : pour spécifier les valeurs de la bordure du boutton
- **Command, CommandParameter** : Valeurs Values for the command to be executed when clicked
- **Font** : Famille de font
- **IsEnabled, IsFocused, IsVisible** : comportement du bouton
- **Text** : label du boutton
- **TextColor** : Couleur du label du boutton

# Button

```
using System;
using Xamarin.Forms;

namespace FormsGallery
{
    class ButtonDemoPage : ContentPage
    {
        Label label;
        int clickTotal = 0;

        public ButtonDemoPage()
        {
            Label header = new Label
            {
                Text = "Button",
                Font = Font.BoldSystemFontOfSize(50),
                HorizontalOptions = LayoutOptions.Center
            };

            Button button = new Button
            {
                Text = "Click Me!",
                Font = Font.SystemFontOfSize(NamedSize.Large),
                BorderWidth = 1,
                HorizontalOptions = LayoutOptions.Center,
                VerticalOptions = LayoutOptions.CenterAndExpand
            };
            button.Clicked += OnButtonClicked;
        }

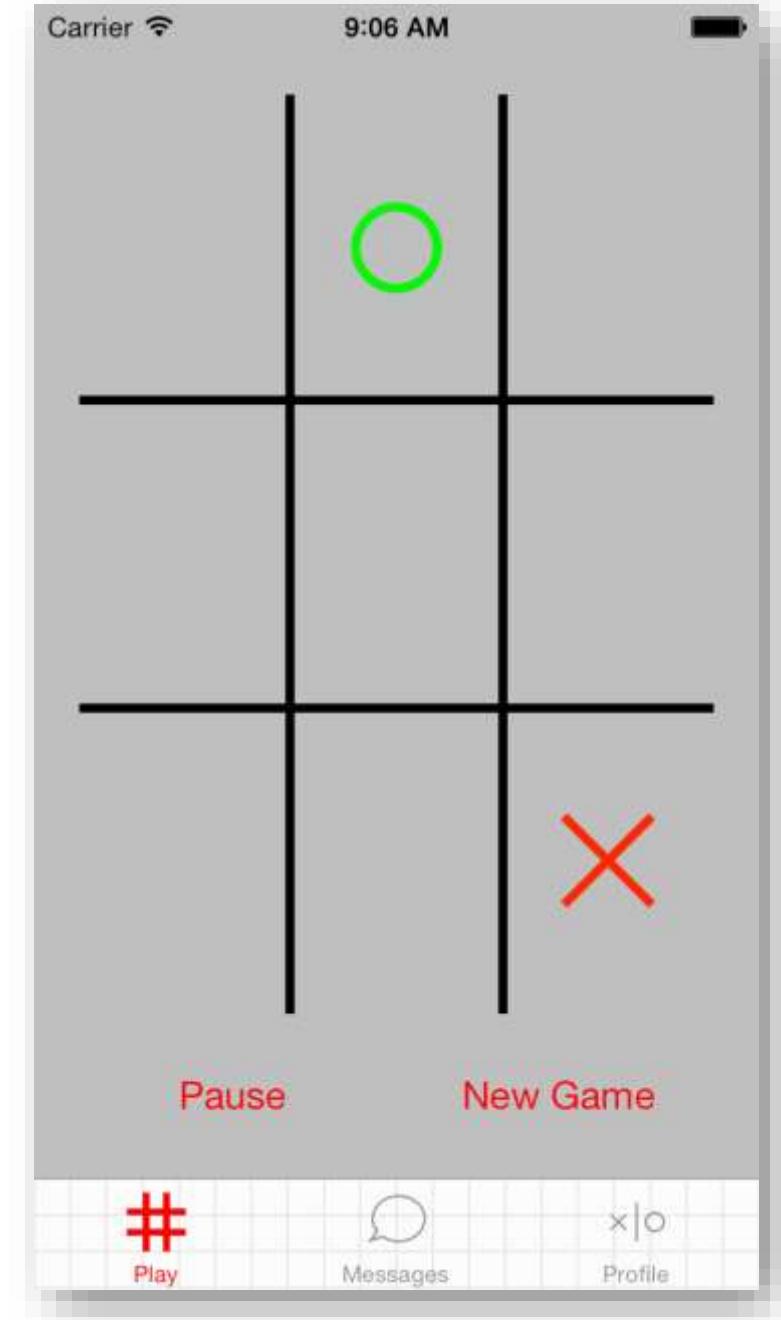
        label = new Label
        {
            Text = "0 button clicks",
            Font = Font.SystemFontOfSize(NamedSize.Large),
            HorizontalOptions = LayoutOptions.Center,
            VerticalOptions = LayoutOptions.CenterAndExpand
        };

        // Accommodate iPhone status bar.
        this.Padding = new Thickness(10, Device.OnPlatform(20, 0, 0), 10, 5);

        // Build the page.
        this.Content = new StackLayout
        {
            Children =
            {
                header,
                button,
                label
            }
        };
    }

    void OnButtonClicked(object sender, EventArgs e)
    {
        clickTotal += 1;
        label.Text = String.Format("{0} button click{1}",
            clickTotal, clickTotal == 1 ? "" : "s");
    }
}
```





# Exercice 10

- Compléter l'exercice de la calculatrice afin de la rendre fonctionnelle
- Implémenter le jeu du TIC-TAC-TOE

# Editor: Réglage et lecture du texte

- L'éditeur, comme d'autres affichages de présentation de texte, propose la propriété Text. Le texte peut être utilisé pour définir et lire le texte présenté par l'éditeur. L'exemple suivant illustre le réglage du texte dans XAML:

```
<Editor Text="I am an Editor" />
```

In C#:

```
var MyEditor = new Editor { Text = "I am an Editor" };
```

To read text, access the Text property in C#:

```
var text = MyEditor.Text;
```

# Entry

- Entry de Xamarin.Forms est utilisée pour la saisie de texte en une seule ligne. Entry, comme la vue Éditeur, prend en charge plusieurs types de clavier. De plus, l'entrée peut être utilisée comme un champ de saisie de mot de passe.
- Entry, comme d'autres vues de présentation de texte, expose la propriété Text. Le texte peut être utilisé pour définir et lire le texte saisi. L'exemple suivant démontre la définition du texte dans XAML :

```
<Entry Text="I am an Entry" />
```

In C#:

```
var MyEntry = new Entry { Text = "I am an Entry" };
```

To read text, access the Text property in C#:

```
var text = MyEntry.Text;
```

## Placeholders

Entry can be set to show placeholder text when it is not storing user input. In practice, this is often seen in forms to clarify the content that is appropriate for a given field. Placeholder text color cannot be customized and will be the same regardless of the TextColor setting. If your design calls for a custom placeholder color, you'll need to fall back to a [custom renderer](#). The following will create an Entry with "Username" as the placeholder in XAML:

```
<Entry Placeholder="Username" />
```

In C#:

```
var MyEntry = new Entry { Placeholder = "Username" };
```

# Entry

- Propriétés:
  - *TextColor* : Color of text and background...can be named color
  - *IsEnabled*, *IsFocused*, *IsPassword*, *IsVisible* : Values that drive entry behaviors
  - *InputTransparent* : Determines whether to show input
  - *Keyboard* : Type of keyboard to show, possible values include **Chat**, **Default**, **Email**, **Numeric**, **Telephone**, **Text**, **Url**
  - *Placeholder* : Text to display when there is no value entered...displayed in grayed out mode
  - *Text* : Value of control

# Entry

## Password Fields

Entry provides the `IsPassword` property. When `IsPassword` is true, the contents of the field will be presented as black circles:

In XAML:

```
<Entry IsPassword="true" />
```

In C#:

```
var MyEntry = new Entry { IsPassword = true };
```



# Entry

Placeholders may be used with instances of Entry that are configured as password fields:

In XAML:

```
<Entry IsPassword="true" Placeholder="Password" />
```

In C#:

```
var MyEntry = new Entry { IsPassword = true, Placeholder = "Password" };
```



# Couleur de l'Entry

Entry can be set to use a custom background and text colors via the following bindable properties:

- `TextColor` – sets the color of the text.
- `BackgroundColor` – sets the color shown behind the text.

Special care is necessary to ensure that colors will be usable on each platform. Because each platform has different defaults for text and background colors, you'll often need to set both if you set one.

Use the following code to set the text color of an entry:

In XAML:

```
<Entry TextColor="Green" />
```

In C#:

```
var entry = new Entry();
entry.TextColor = Color.Green;
```



# Picker

- Picker est un bon choix lorsque vous avez besoin d'une vue qui permet à l'utilisateur de choisir un élément parmi une petite collection de plusieurs éléments. Picker est implémenté de manière spécifique à la plate-forme et à la limitation que chaque élément est identifié uniquement par une chaîne de texte.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="PickerDemo.PickerDemoPage">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness"
            iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <StackLayout Padding="20">
        <Entry x:Name="entry"
            Placeholder="Type something, type anything" />
        <Picker Title="Keyboard Type"
            SelectedIndexChanged="OnPickerSelectedIndexChanged">
            <Picker.Items>
                <x:String>Default</x:String>
                <x:String>Text</x:String>
                <x:String>Chat</x:String>
                <x:String>Url</x:String>
                <x:String>Email</x:String>
                <x:String>Telephone</x:String>
                <x:String>Numeric</x:String>
            </Picker.Items>
        </Picker>
    </StackLayout>
</ContentPage>
```

Le programme définit deux propriétés de **Picker**: la propriété **Title** est une chaîne qui identifie la fonction du **Picker**. La propriété **Items** est du type **IList <string>**, et généralement vous l'initialisez avec une liste de **x: String** tags dans le fichier XAML. (Picker n'a pas d'attribut de propriété de contenu, donc les étiquettes explicites de **Picker.Items** sont requises.) Dans le code, vous pouvez utiliser la méthode **Ajouter** ou **Insérer** définie par "**IList <string>**" pour mettre des éléments de chaîne dans la collection.



# Picker

Le programme PickerDemo gère l'événement SelectedIndexChanged dans le fichier code-behind. Il obtient SelectedIndex du Picker, utilise ce nombre pour indexer la collection Items du Picker, puis utilise la réflexion pour obtenir l'objet Keyboard correspondant, qu'il définit sur la propriété Keyboard de l'entrée:

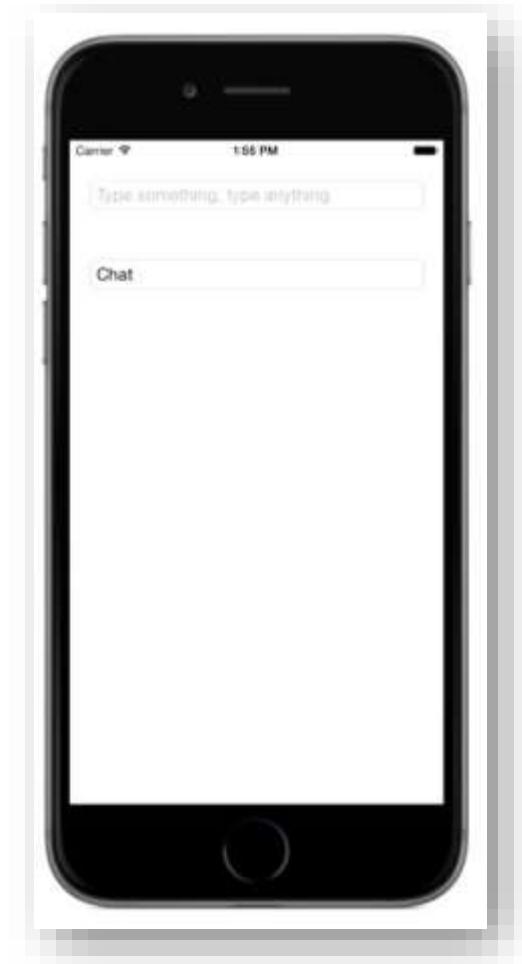
```
public partial class PickerDemoPage : ContentPage
{
    public PickerDemoPage()
    {
        InitializeComponent();
    }

    void OnPickerSelectedIndexChanged(object sender, EventArgs args)
    {
        if (entry == null)
            return;

        Picker picker = (Picker)sender;
        int selectedIndex = picker.SelectedIndex;

        if (selectedIndex == -1)
            return;

        string selectedItem = picker.Items[selectedIndex];
        PropertyInfo propertyInfo = typeof(Keyboard).GetRuntimeProperty(selectedItem);
        entry.Keyboard = (Keyboard)propertyInfo.GetValue(null);
    }
}
```



# Switch

Les programmes d'application ont souvent besoin d'une entrée booléenne de l'utilisateur, ce qui requiert que l'utilisateur permette d'activer ou de désactiver une option de programme, oui ou non, vrai ou faux, ou si vous voulez en penser. Dans Xamarin.Forms, c'est une vue appelée Switch.

- Switch définit une propriété seule, nommée `IsToggled` de type boolean, et elle déclenche l'événement `Toggled` pour indiquer une modification de cette propriété.
- Dans le code, vous pourriez être enclin à donner à un Switch un nom de commutateur, mais c'est un mot-clé C #, donc vous voudrez choisir autre chose. Toutefois, dans XAML, vous pouvez définir l'attribut `x: Name` à changer, et l'analyseur XAML créera intelligemment un champ nommé `@switch`, de sorte que C # vous permet de définir un nom de variable à l'aide d'un mot-clé C #.
- *IsToggled, IsEnabled, IsVisible, IsFocused* : Drive control's behaviors



# Switch

Le programme SwitchDemo crée deux éléments Switch avec deux étiquettes d'identification: "Italic" et "Boldface". Chaque commutateur a son propre gestionnaire d'événements, qui forme l'étiquette plus grande au bas du StackLayout:

Le gestionnaire d'événements Toggled possède un deuxième argument de ToggledEventArgs, qui possède une propriété Value du type bool qui indique le nouvel état de la propriété IsToggled. Les gestionnaires d'événements dans SwitchDemo utilisent cette valeur pour définir ou effacer l'indicateur FontAttributes en particulier dans la propriété FontAttributes de l'étiquette longue:

```
public partial class SwitchDemoPage : ContentPage
{
    public SwitchDemoPage()
    {
        InitializeComponent();
    }

    void OnItalicSwitchToggled(object sender, ToggledEventArgs args)
    {
        if (args.Value)
        {
            label.FontAttributes |= FontAttributes.Italic;
        }
        else
        {
            label.FontAttributes &= ~FontAttributes.Italic;
        }
    }

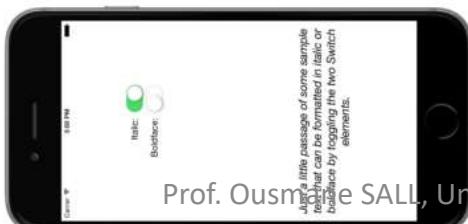
    void OnBoldSwitchToggled(object sender, ToggledEventArgs args)
    {
        if (args.Value)
        {
            label.FontAttributes |= FontAttributes.Bold;
        }
        else
        {
            label.FontAttributes &= ~FontAttributes.Bold;
        }
    }
}
```

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="SwitchDemo.SwitchDemoPage">

    <StackLayout Padding="10, 0">
        <StackLayout HorizontalOptions="Center"
                    VerticalOptions="CenterAndExpand">
            <StackLayout Orientation="Horizontal"
                        HorizontalOptions="End">
                <Label Text="Italic: "
                      VerticalOptions="Center" />
                <Switch Toggled="OnItalicSwitchToggled"
                        VerticalOptions="Center" />
            </StackLayout>

            <StackLayout Orientation="Horizontal"
                        HorizontalOptions="End">
                <Label Text="Boldface: "
                      VerticalOptions="Center" />
                <Switch Toggled="OnBoldSwitchToggled"
                        VerticalOptions="Center" />
            </StackLayout>
        </StackLayout>
    </StackLayout>

    <Label x:Name="label"
          Text="Just a little passage of some sample
text that can be formatted in italic or boldface by toggling the two Switch elements."
          FontSize="Large"
          HorizontalTextAlignment="Center"
          VerticalOptions="CenterAndExpand" />
</StackLayout>
</ContentPage>
```



# Exercice 11

- Créer une application permettant une conversion monétaire du CFA à l’Euro et respectivement de l’Euro au CFA (à noter 01 Euro=655,55 F CFA). Cette application comprendra un Picker permettant de choisir le sens de la conversion.

# Exercice 12

- Créer une application permettant une conversion une conversion de Km à Miles (à noter 01 KM=0,62 Miles) et respectivement de Miles au KM. Cette application comprendra un Picker permettant de choisir le sens de la conversion.

# Exercice 13

- Créer une fenêtre avec une rangée de 3 boutons ("Couleur1", "Couleur 2 ", et " Couleur 3 ") dans la partie supérieure. Le troisième bouton doit remplir tout l'espace restant. Dans la ligne suivante, insérer une label occupant toute la largeur et ayant une hauteur fixe. La troisième rangée doit contenir un seul bouton ("Effacer") à placer à droite.

# DatePicker

- *Propriétés:*

- *Date* : The date value of the control
- *Format* : Format to display, using standard C# formats

- La représentation visuelle d'un DatePicker est très similaire à celle de Entry, sauf qu'un contrôle spécial pour choisir une date apparaît à la place d'un clavier.

#### C# Example

```
DatePicker datePicker = new DatePicker
{
    Format = "D",
    VerticalOptions = LayoutOptions.CenterAndExpand
};
```

#### XAML Example

```
<StackLayout>
    <DatePicker VerticalOptions="CenterAndExpand" Date="{x:Static sys:DateTime.Now}">
        <DatePicker.Format>yyyy-MM-dd</DatePicker.Format>
        <DatePicker.MinimumDate>
            <sys:DateTime x:FactoryMethod="Parse">
                <x:Arguments>
                    <x:String>Jan 1 2000</x:String>
                </x:Arguments>
            </sys:DateTime>
        </DatePicker.MinimumDate>
        <DatePicker.MaximumDate>
            <sys:DateTime x:FactoryMethod="Parse">
                <x:Arguments>
                    <x:String>Dec 31 2050</x:String>
                </x:Arguments>
            </sys:DateTime>
        </DatePicker.MaximumDate>
    </DatePicker>
</StackLayout>
```

# DatePicker

XAML for Xamarin.Forms supports the following properties for the [DatePicker](#) class:

PROPERTY	VALUE
Format	A string that specifies the display format in the control of the chosen date.
Date	An <code>x:FactoryMethod</code> call to the <code>DateTime.Parse</code> method, or a markup extension call to a method that produces a <code>DateTime</code> object. See below.
MinimumDate	An <code>x:FactoryMethod</code> call to the <code>DateTime.Parse</code> method, or a markup extension call to a method that produces a <code>DateTime</code> object. See below.
MaximumDate	An <code>x:FactoryMethod</code> call to the <code>DateTime.Parse</code> method, or a markup extension call to a method that produces a <code>DateTime</code> object. See below.

The example below creates a working [DatePicker](#) that displays the current date and allows the user to select a date between the specified ranges. The value for the `DatePicker.Date` property is specified with the `x:Static` markup extension, and the `DatePicker.MinimumDate` and `DatePicker.MaximumDate` properties are specified by calling the `DateTime.Parse` method with the `x:FactoryMethod` and `x:Arguments` tags.

*Note:* The example below requires a namespace declaration in the root `ContentPage` or `ContentView` tags. In particular, `xmlns:sys="clr-namespace:System;assembly=mscorlib"` must appear in the attribute list for the root element, so that the XAML parser can resolve the name, `sys:DateTime`.

# DatePicker

## Public Constructors

<a href="#">DatePicker()</a>	Initializes a new instance of the DatePicker class.
------------------------------	---

## Public Fields

static readonly	<a href="#">DateProperty</a>	BindableProperty. Identifies the Date bindable property.
static readonly	<a href="#">FormatProperty</a>	BindableProperty. Identifies the Format dependency property.
static readonly	<a href="#">MaximumDateProperty</a>	BindableProperty. Identifies the MaximumDate bindable property.
static readonly	<a href="#">MinimumDateProperty</a>	BindableProperty. Identifies the MinimumDate bindable property.
static readonly	<a href="#">TextColorProperty</a>	BindableProperty. Backing store for the <code>DatePicker.TextColor</code> property.

# DatePicker

## Public Properties

<a href="#">Date</a>	<code>DateTime</code> . Gets or sets the displayed date. This is a bindable property.
<a href="#">Format</a>	<code>String</code> . The format of the date to display to the user. This is a dependency property.
<a href="#">MaximumDate</a>	<code>DateTime</code> . The highest date selectable for this DatePicker. This is a bindable property.
<a href="#">MinimumDate</a>	<code>DateTime</code> . The lowest date selectable for this DatePicker. This is a bindable property.
<a href="#">TextColor</a>	<code>Color</code> . Gets or sets the text color for the date picker.

## Public Methods

[On<T>\(\)](#) : `IPlatformElementConfiguration<T, DatePicker>`

Returns the platform-specific instance of this `DatePicker`, on which a platform-specific method may be called.

## Public Events

<a href="#">DateSelected</a>	An event fired when the Date property changes.
------------------------------	--

# TimePicker

# ListView

- Composant vous permettant d'afficher des éléments sous forme de liste de taille (presque) infinie
- Très populaire dans les applications mobiles
- Peut être relié dans votre ViewModel à :
  - Une `List<T>` => si la liste n'est pas amenée à changer (pas d'ajout/suppression)
  - Une `ObservableCollection<T>` => si vous souhaitez ajouter/supprimer des éléments dans la collection
- Propriétés:
  - *ItemsSource* : The data collection used to populate the list
  - *SelectedItem* : If an item is selected in the list
  - *Enabled*, *GroupingEnabled*, *Visible* : Drives list's behaviors
  - *RowHeight* : Height of item's row
  - *HasUnevenRows* : A flag that indicates row height varies
  - *Triggers* : connects an item to a behavior
  - *ItemTemplate* : Binding template for item's display
  - *GroupHeaderTemplate* : Binding template for list's header

# ListView

- Propriétés importantes
  - **ItemSource** => à relier à la collection de votre ViewModel
  - **SelectedItem** => vous permet de récupérer l'élément sélectionné dans votre ViewModel
  - **ItemTemplate** => définit la manière d'afficher chaque élément de la liste

```
<ListView ItemsSource="{Binding Items}"
          SelectedItem="{Binding SelectedItem, Mode=TwoWay}"
          ItemTemplate="{StaticResource ListTemplate}"
          />

<ContentPage.Resources>
    <ResourceDictionary>
        <DataTemplate x:Key="ListTemplate">
            <ViewCell>
                <Label Text="{Binding Title}"
                      FontSize="20"
                      />
            </ViewCell>
        </DataTemplate>
    </ResourceDictionary>
</ContentPage.Resources>
```

# ListView : Evenement

- Comment réagir quand l'utilisateur sélectionne un élément ?
- **Solution 1** : La ListView dispose d'un event ItemSelected.
  - Problème : on ne peut pas lier un événement avec MVVM !
- **Solution 2** : Utiliser un Binding TwoWay sur le SelectedItem

```
SelectedItem="{Binding SelectedItem, Mode=TwoWay}"  
  
public TodoItem SelectedItem  
{  
    get { return _selectedItem; }  
    set  
    {  
        if ( SetProperty<TodoItem>(ref _selectedItem, value))  
        {  
            if (value != null)  
            {  
                OnItemSelected(value);  
            }  
        }  
    }  
}
```

# Map in Xamarin.Forms

- Xamarin.Forms.Maps utilise les Map API natives sur chaque plate-forme. Cela fournit une expérience de cartes rapide et familière pour les utilisateurs, mais cela signifie que certaines étapes de configuration sont nécessaires pour respecter les exigences spécifiques de chaque plate-forme API. Une fois configuré, le contrôle Map fonctionne comme n'importe quel autre élément Xamarin.Forms dans le code commun.

# Maps Initialization

Lorsque vous ajoutez des cartes à une application Xamarin.Forms, Xamarin.Forms.Maps est un paquet NuGet distinct que vous devez ajouter à chaque projet dans la solution. Sur Android, cela a également une dépendance sur GooglePlayServices (un autre NuGet) qui est automatiquement téléchargé lorsque vous ajoutez Xamarin.Forms.Maps.

- Après avoir installé le paquet NuGet, un certain code d'initialisation est nécessaire dans chaque projet d'application, après l'appel de la méthode Xamarin.Forms.Forms.Init. Pour iOS, utilisez le code suivant:

```
Xamarin.FormsMaps.Init();
```

On Android you must pass the same parameters as Forms.Init:

```
Xamarin.FormsMaps.Init(this, bundle);
```

For the Windows Runtime (WinRT) and the Universal Windows Platform (UWP) use the following code:

```
Xamarin.FormsMaps.Init("INSERT_AUTHENTICATION_TOKEN_HERE");
```

# Configuration de la plate-forme: Android

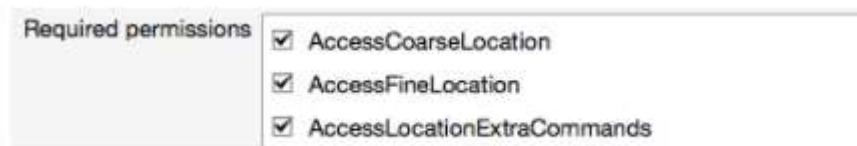
Pour utiliser Google Maps API v2 sur Android, vous devez générer une clé API et l'ajouter à votre projet Android. Suivez les instructions dans le document Xamarin lors de l'obtention d'une clé Google Maps API v2. Après avoir suivi ces instructions, collez la clé API dans le fichier **Properties/AndroidManifest.xml** (voir la source et trouver / mettre à jour l'élément suivant:

```
<meta-data android:name="com.google.android.maps.v2.API_KEY"
    android:value="AbCdEfGhIjKlMnOpQrStUvWValueGoesHere" />
```

N'oubliez pas de générer une autre clé en utilisant le fichier **keystore** qui est utilisé pour signer la version Release de toute application qui est téléchargée sur Google Play Store. La clé que vous générez pour le développement et le débogage ne fonctionnera pas et l'application téléchargée à partir de Google Play affichera une carte défectueuse. N'oubliez pas de régénérer la clé si le nom du package de l'application change.

Vous devrez également activer les autorisations appropriées en cliquant avec le bouton droit de la souris sur le projet Android et en sélectionnant **Options > Build > Android Application** et cocher la case suivante :

- AccessCoarseLocation
- AccessFineLocation
- AccessLocationExtraCommands
- AccessMockLocation
- AccessNetworkState
- AccessWifiState
- Internet



# Using Maps

```
public class MapPage : ContentPage {
    public MapPage() {
        var map = new Map(
            MapSpan.FromCenterAndRadius(
                new Position(37,-122), Distance.FromMiles(0.3))) {
            IsShowingUser = true,
            HeightRequest = 100,
            WidthRequest = 960,
            VerticalOptions = LayoutOptions.FillAndExpand
        );
        var stack = new StackLayout { Spacing = 0 };
        stack.Children.Add(map);
        Content = stack;
    }
}
```

# ProgressBar

- Used to display how far along a process is
  - *IsEnabled*, *IsFocused*, *IsVisible* : Values to drive the control's behaviors
  - *Progress* : Value to show completion...I think this is a value from 0.0 to 1.0

# SearchBar

- Le SearchBar ne provient pas de InputView comme Entry and Editor, et il n'a pas de propriété Keyboard. Le clavier que SearchBar affiche lorsqu'il acquiert le focus d'entrée est spécifique à la plate-forme et approprié pour une commande de recherche. Le SearchBar lui-même est similaire à une vue d'entrée, mais selon la plate-forme, il peut être orné de certains autres graphiques et contenir un bouton qui efface le texte saisi.
- SearchBar définit deux événements:
  - TextChanged
  - SearchButtonPressed
- SearchBar définit cinq propriétés:
  - Text - le texte saisi par l'utilisateur
  - Placeholder - texte d'avis affiché avant que l'utilisateur commence à taper
  - CancelButtonColor - de type Couleur
  - SearchCommand - pour utilisation avec liaison de données
  - SearchCommandParameter - pour utilisation avec la liaison de données



# SearchBar

Le programme **SearchBarDemo** utilise seulement **Text** et **Placeholder**, mais le fichier XAML attache les gestionnaires pour les deux événements :

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="SearchBarDemo.SearchBarDemoPage">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness"
            iOS="10, 20, 10, 0"
            Android="10, 0"
            WinPhone="10, 0" />
    </ContentPage.Padding>

    <StackLayout>
        <earchBar x:Name="searchBar"
            Placeholder="Search text"
            TextChanged="OnSearchBarTextChanged"
            SearchButtonPressed="OnSearchBarButtonPressed" />

        <ScrollView x:Name="resultsScroll"
            VerticalOptions="FillAndExpand">
            <StackLayout x:Name="resultsStack" />
        </ScrollView>
    </StackLayout>
</ContentPage>
```

# Slider et Stepper

- Slider et Stepper permettent à l'utilisateur de sélectionner une valeur numérique à partir d'une plage. Ils ont des interfaces de programmation presque identiques, mais incorporent des paradigmes visuels et interactifs très différents.
- Le curseur Xamarin.Forms est une barre horizontale qui représente une gamme de valeurs entre un minimum à gauche et un maximum à droite. (Le curseur Xamarin.Forms ne prend pas en charge une orientation verticale).
- Le curseur définit trois propriétés publiques de type double, nommées Minimum, Maximum et Valeur. Chaque fois que la propriété Value change, le Slider déclenche un événement ValueChanged indiquant la nouvelle valeur.
- Lorsque vous affichez un curseur, vous voudrez un peu de rembourrage à gauche et à droite pour éviter que le curseur ne s'étende jusqu'aux bords de l'écran.

# Slider et Stepper

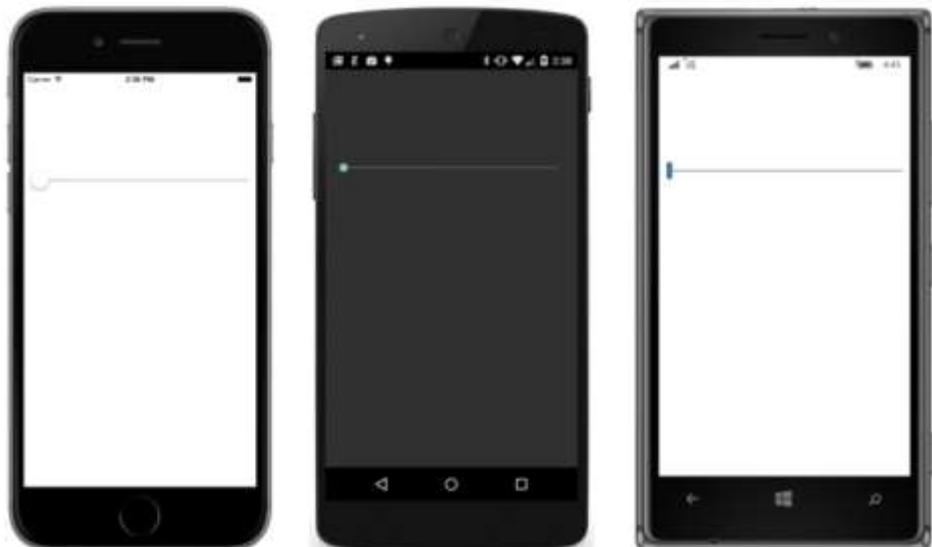
- Le fichier XAML dans le programme **SliderDemo** applique le Padding à StackLayout, qui est parent à la fois d'un Slider et d'un Label qui est destiné à afficher la valeur actuelle du Slider:

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="SliderDemo.SliderDemoPage">

    <StackLayout Padding="10, 0">
        <Slider VerticalOptions="CenterAndExpand"
            ValueChanged="OnSliderValueChanged" />

        <Label x:Name="label"
            FontSize="Large"
            HorizontalOptions="Center"
            VerticalOptions="CenterAndExpand" />
    </StackLayout>
</ContentPage>
```

Lorsque le programme démarre, l'étiquette ne présente rien, et le curseur est positionné à l'extrême gauche:



# Slider et Stepper

- Le Slider avise le code de modifications dans la propriété Value en déclenchant l'événement ValueChanged. L'événement est déclenché si la valeur est modifiée par programmation ou par la manipulation de l'utilisateur. Voici le fichier SliderDemo code-behind avec le

```
public partial class SliderDemoPage : ContentPage
{
    public SliderDemoPage()
    {
        InitializeComponent();
    }

    void OnSliderValueChanged(object sender, ValueChangedEventArgs args)
    {
        label.Text = String.Format("Slider = {0}", args.NewValue);
    }
}
```



# TableView

# WebView: display webpages or HTML

The `WebView` is intended to embed a web browser inside your application. Alternatively, you can use `WebView` in conjunction with the `HtmlWebViewSource` class to display a chunk of HTML, perhaps saved as an embedded resource in the PCL.

Here's the XAML file for **WebViewDemo**. Notice that the nested `StackLayout` containing the two `Button` elements has its `BindingContext` property set to the `WebView`. The two `Button` children in that `StackLayout` inherit the `BindingContext`, so the buttons can have very simple Binding expressions on their `IsEnabled` properties that reference only the `CanGoBack` and `CanGoForward`.

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="WebViewDemo.WebViewDemoPage">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness"
            iOS="10, 20, 10, 0"
            Android="10, 0"
            WinPhone="10, 0" />
    </ContentPage.Padding>

    <StackLayout>
        <Entry Keyboard="Url"
            Placeholder="web address"
            Completed="OnEntryCompleted" />
        <StackLayout Orientation="Horizontal"
            BindingContext="{x:Reference webView}">
            <Button Text=" ">
                <Button.HorizontalOptions>FillAndExpand</Button.HorizontalOptions>
                <Button.IsEnabled>{Binding CanGoBack}</Button.IsEnabled>
                <Button.Clicked>OnGoBackClicked</Button.Clicked>
            </Button>
            <Button Text=" ">
                <Button.HorizontalOptions>FillAndExpand</Button.HorizontalOptions>
                <Button.IsEnabled>{Binding CanGoForward}</Button.IsEnabled>
                <Button.Clicked>OnGoForwardClicked</Button.Clicked>
            </Button>
        </StackLayout>
    </StackLayout>
    <WebView x:Name="webView"
        VerticalOptions="FillAndExpand"
        Source="https://xamarin.com" />
</ContentPage>
```

`WebView`

A `View` that presents HTML content.  
[WebView API](#)  
[WebView documentation](#)  
[Demo source](#)



# WebView: display webpages or HTML

The code-behind file needs to handle the `Clicked` events for the **Back** and **Forward** buttons as well as the `Completed` event for the `Entry` that lets you enter a web address of your own:

```
public partial class WebViewDemoPage : ContentPage
{
    public WebViewDemoPage()
    {
        InitializeComponent();
    }

    void OnEntryCompleted(object sender, EventArgs args)
    {
        webView.Source = ((Entry)sender).Text;
    }

    void OnGoBackClicked(object sender, EventArgs args)
    {
        webView.GoBack();
    }

    void OnGoForwardClicked(object sender, EventArgs args)
    {
        webView.GoForward();
    }
}
```



# TextCell

- Displays text and detail subtext in a single control
  - *Text* : Le texte principal à afficher
  - *TextColor* : Couleur primaire du texte
  - *Detail* : Subtext displayed below main text
  - *DetailColor* : Color of secondary text

# ImageCell

- Displays an image and a TextCell
  - *ImageSource*
  - *Text, TextColor*
  - *Detail, DetailColor*
  - *IsEnabled*

# Exercice 14

- L'objectif de cet exercice est d'implémenter un jeu entre l'utilisateur et l'ordinateur.
- **Processus:**
  - L'utilisateur choisit entre caillou, papier or ciseau
  - Ordinateur: générer un choix parmi les trois
  - Regarder qui a gagné et afficher le résultat
  - Compter le nombre de fois que chacun à gagné

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 5312 Technologies Mobiles: Développement d'Applications Mobiles Cross-Platform avec Xamarin et C#

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

**Xamarin - Data-Binding - Model-View-ViewModel(MVVM)**

# A propos de moi

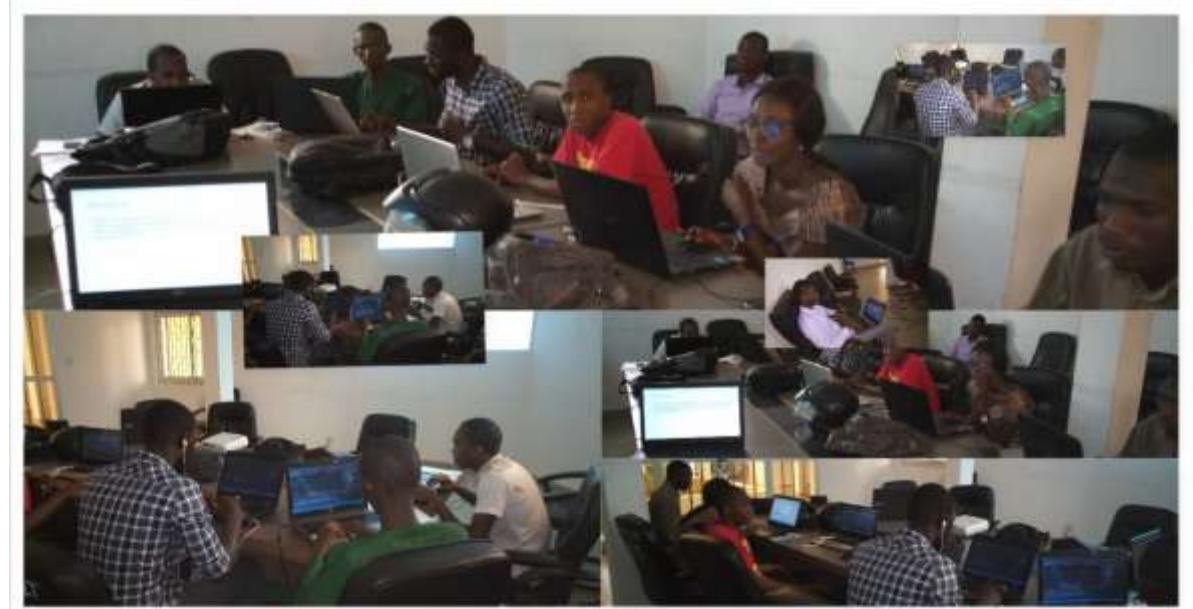


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

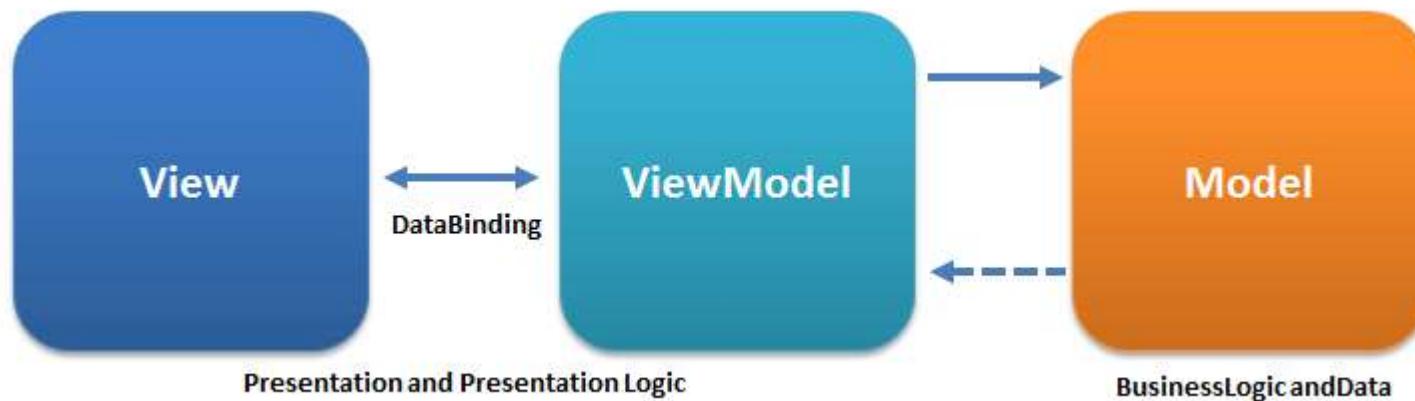
1. Une architecture pour le développement Cross-Plateforme
2. Développement Cross-Platform avec Xamarin.Forms
3. Interface utilisateur
- 4. Data-Binding**
5. Web Services
6. Test et Déploiement

# Introduction au binding

- MVVM permet de lier les données à l'interface graphique. Sorte de pattern de développement qui permet de séparer la couche logique et l'IHM(sorte de MVC pour les applications XAML).
- Cette liaison s'effectue à l'aide d'un mécanisme appelé « Data-Binding »
- Cette liaison peut être:
  - De la donnée vers l'interface(Affichage dans un Label)
  - De l'Interface vers la donnée(Affichage dans une zone de texte)
  - Les 2( donnée--> interface et interface-->donnée)

# Qu'est ce que c'est ?

- Model-View-ViewModel



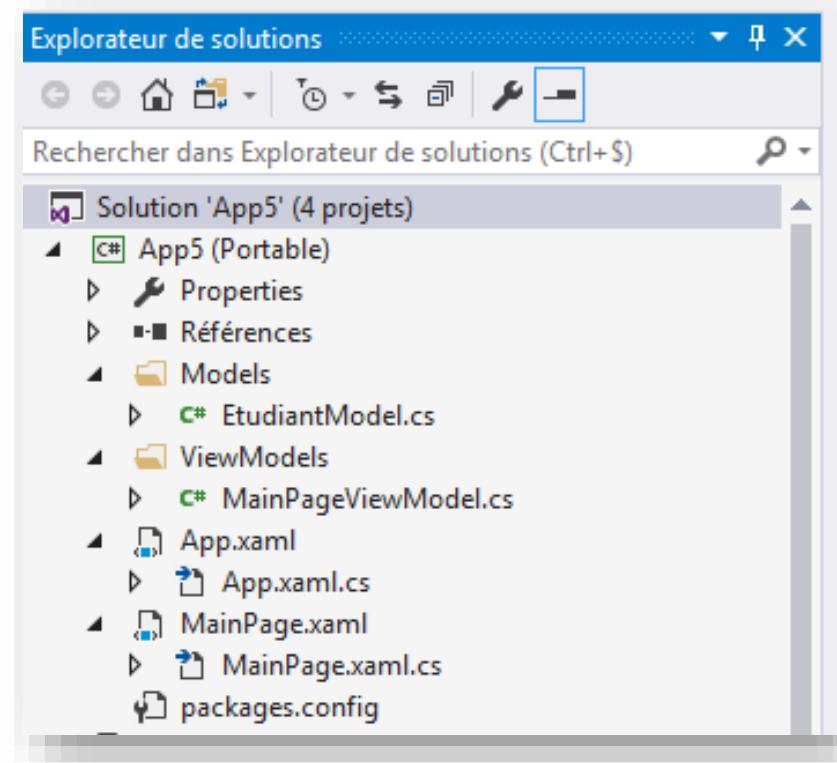
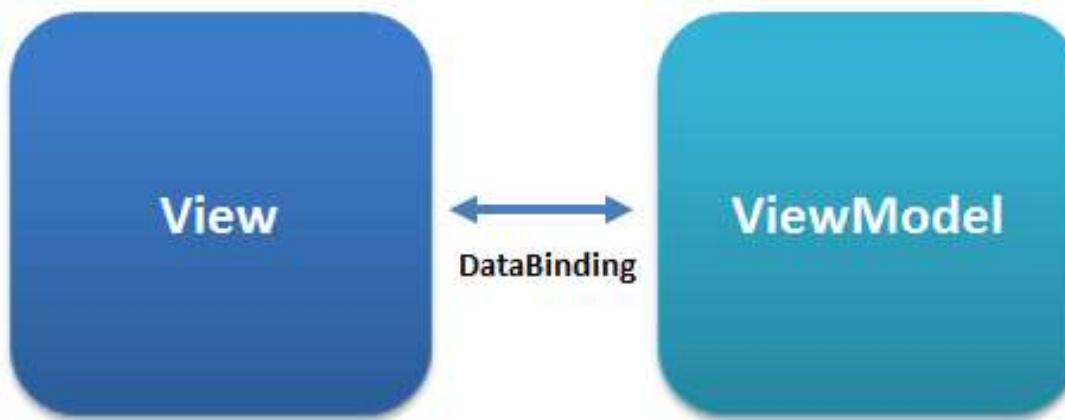
- **Model/ViewModel => C#**
- **View => Xaml**

- **Model:** Représente les données qui seront affichées et manipulées par l'utilisateur;
- **View:** L'interface graphique de l'application
- **ViewModel:** Fait le lien entre les données, l'interface graphique et l'utilisateur.

# Quel intérêt ?

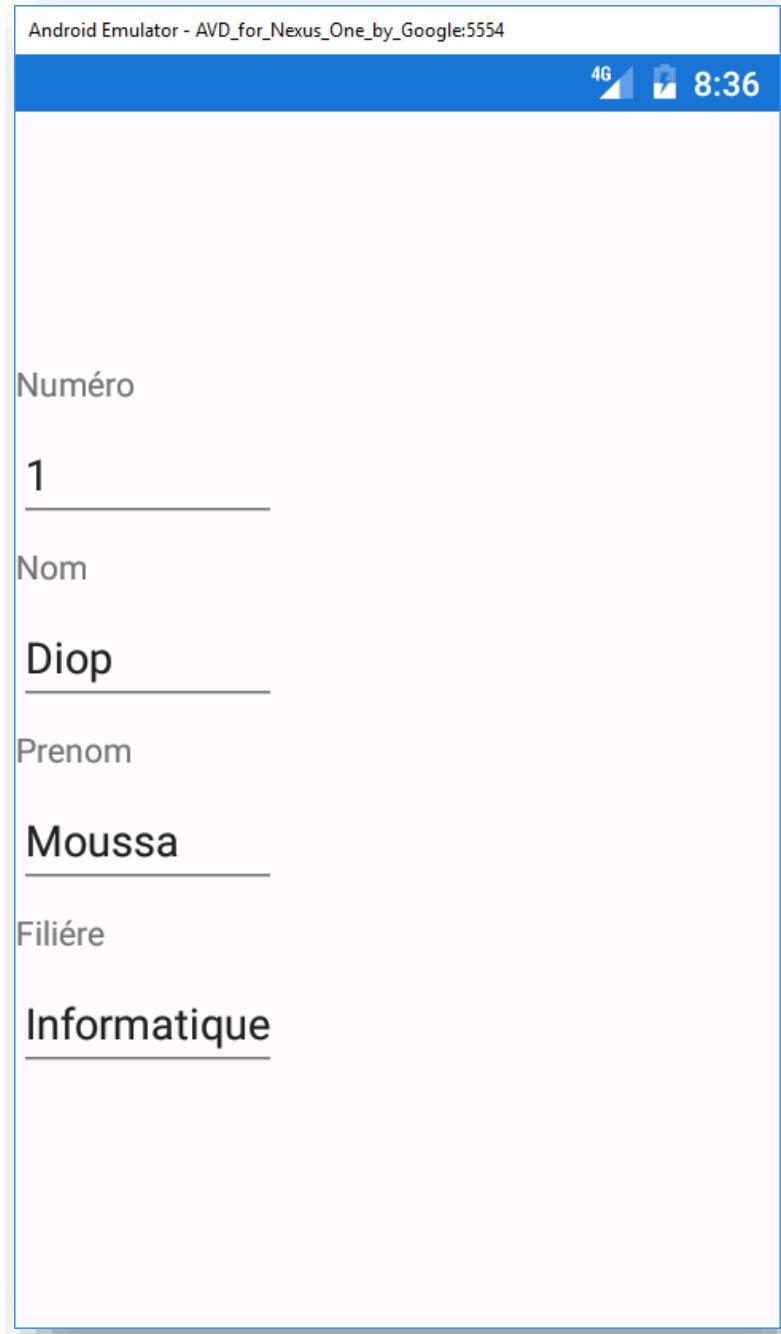
- Couplage faible entre Vue et ViewModel
- Développement dissocié
  - => Plus simple de gérer le développement en équipe
- Le designer développe la Vue,
- Le développeur le reste ☺

# Comment communiquer ?



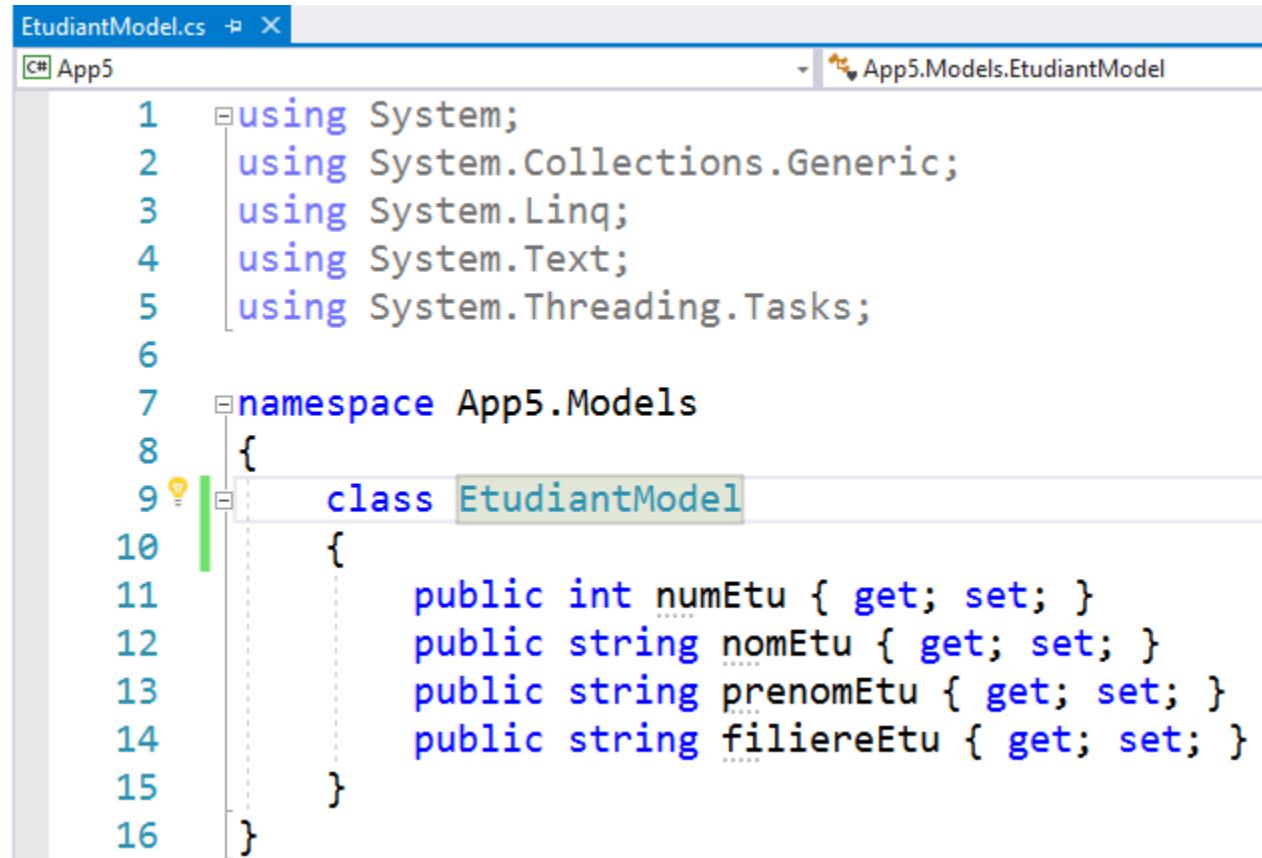
- Moteur de binding
- La vue « demande » qu'on lie certaines propriétés du ViewModel
  - Propriété « simple » => Récupération de valeur pour l'affichage
  - Command => Lien pour que le ViewModel réagisse sur les actions de la Vue

# Comment communiquer ?



# Comment communiquer ?

- EtudiantModel.cs

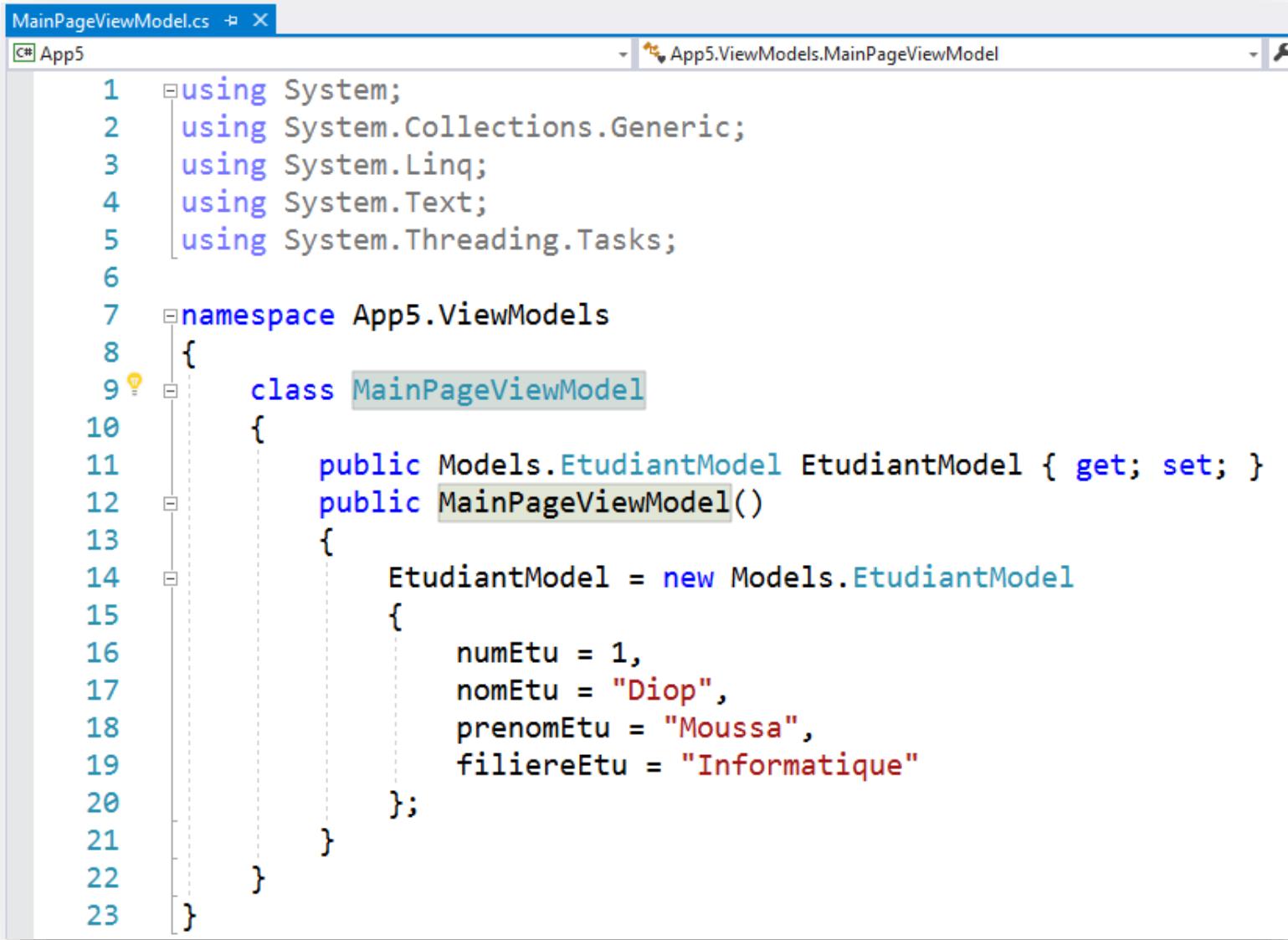


The screenshot shows a code editor window with the title bar "EtudiantModel.cs" and a tab "App5". The namespace "App5.Models" is selected. The code defines a class "EtudiantModel" with properties: numEtu, nomEtu, prenomEtu, and filiereEtu, each with get and set accessors.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace App5.Models
8  {
9      class EtudiantModel
10     {
11         public int numEtu { get; set; }
12         public string nomEtu { get; set; }
13         public string prenomEtu { get; set; }
14         public string filiereEtu { get; set; }
15     }
16 }
```

# Comment communiquer ?

- MainPageViewModel.cs

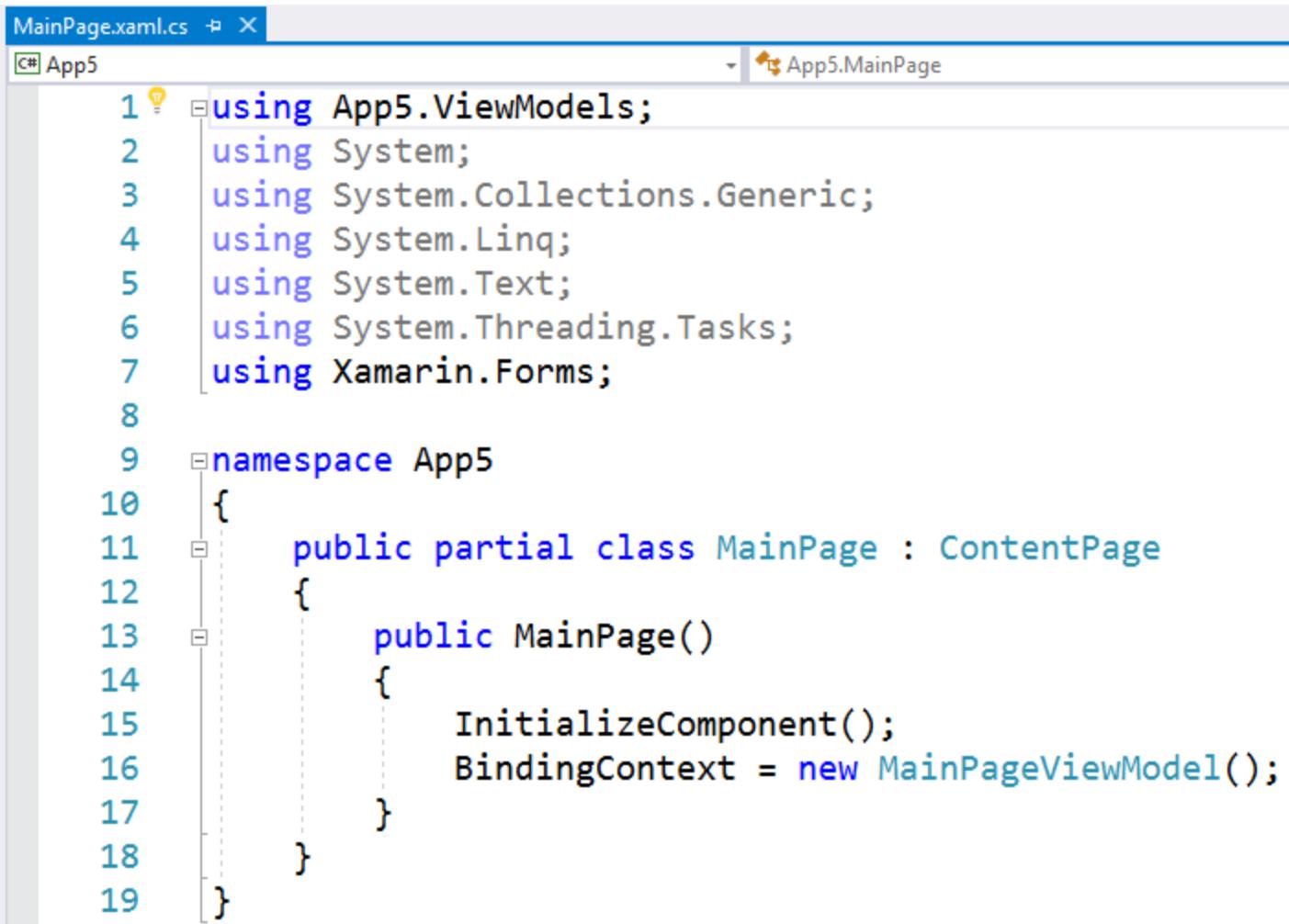


The screenshot shows a Microsoft Visual Studio code editor window titled "MainPageViewModel.cs". The code is written in C# and defines a class named "MainPageViewModel". The class contains a public property "EtudiantModel" and a constructor that initializes an instance of "EtudiantModel" with specific values: numEtu = 1, nomEtu = "Diop", prenomEtu = "Moussa", and filiereEtu = "Informatique".

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace App5.ViewModels
8  {
9      class MainPageViewModel
10     {
11         public Models.EtudiantModel EtudiantModel { get; set; }
12         public MainPageViewModel()
13         {
14             EtudiantModel = new Models.EtudiantModel
15             {
16                 numEtu = 1,
17                 nomEtu = "Diop",
18                 prenomEtu = "Moussa",
19                 filiereEtu = "Informatique"
20             };
21         }
22     }
23 }
```

# Comment communiquer ?

- MainPage.cs

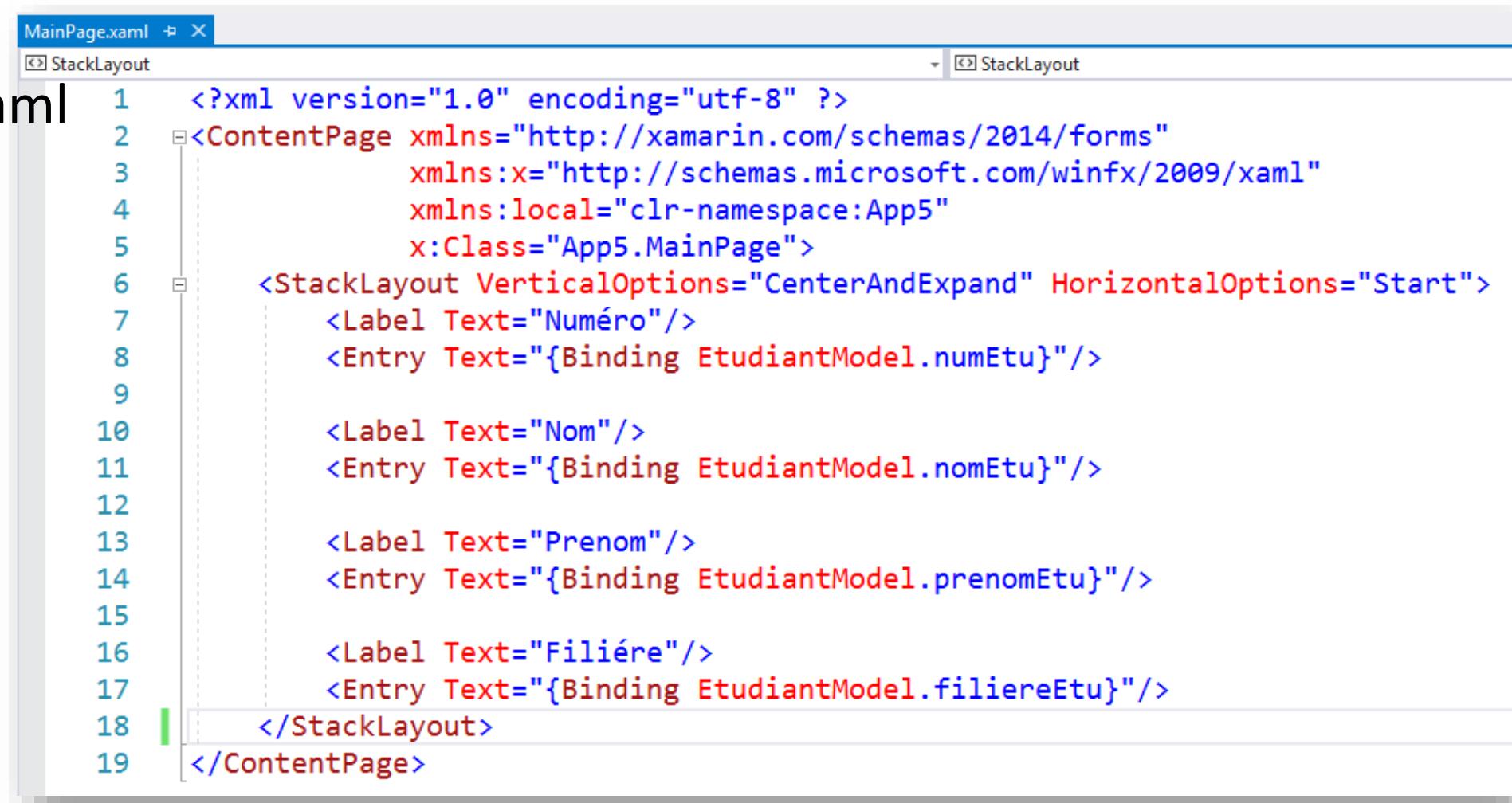


The screenshot shows a code editor window for a C# file named MainPage.xaml.cs. The title bar indicates the file name and the project name App5. The code itself is as follows:

```
1  using App5.ViewModels;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using Xamarin.Forms;
8
9  namespace App5
10 {
11     public partial class MainPage : ContentPage
12     {
13         public MainPage()
14         {
15             InitializeComponent();
16             BindingContext = new MainPageViewModel();
17         }
18     }
19 }
```

# Comment communiquer ?

- MainPage.xaml



The screenshot shows the MainPage.xaml file in a Xamarin Studio code editor. The window title is "MainPage.xaml". The code is XAML and defines a ContentPage with a StackLayout containing four pairs of Label and Entry controls, each bound to properties of an EtudiantModel object.

```
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      xmlns:local="clr-namespace:App5"
5      x:Class="App5.MainPage">
6      <StackLayout VerticalOptions="CenterAndExpand" HorizontalOptions="Start">
7          <Label Text="Numéro"/>
8          <Entry Text="{Binding EtudiantModel.numEtu}"/>
9
10         <Label Text="Nom"/>
11         <Entry Text="{Binding EtudiantModel.nomEtu}"/>
12
13         <Label Text="Prenom"/>
14         <Entry Text="{Binding EtudiantModel.prenomEtu}"/>
15
16         <Label Text="Filière"/>
17         <Entry Text="{Binding EtudiantModel.filiereEtu}"/>
18     </StackLayout>
19 </ContentPage>
```

# Comment communiquer ?

- Utilisation des bindings pour récupérer les valeurs du ViewModel

```
<Label Text="Nom"/>  
<Entry Text="{Binding EtudiantModel.nomEtu}"/>
```

```
public string nomEtu { get; set; }
```



# Comment utiliser les Bindings ?

- {**Binding** NomPropriété [, attribut=valeur] }
- *Les attributs*
- Mode :
  - **OneWay** : met à jour du ViewModel vers la Vue
  - **OneWayToSource** : met à jour de la Vue vers le ViewModel
  - **TwoWay** : met à jour dans les deux sens
  - **Default** : met le mode par défaut (dépendant des propriétés)
- **StringFormat** : format pour afficher la valeur (@see string.Format)
  - Example : "{Binding Age, StringFormat='Vous avez {0} ans'}«

# Comment utiliser les Bindings ?

- Converter : permet un prétraitement des données avant de les afficher
- Un converter est une classe C# qui implémente `IValueConverter`
  - `Convert` (`ViewModel` -> `Vue`)
  - `ConvertBack` (`Vue` -> `ViewModel`)
- Exemple : `StringToUpperConverter.cs`

```
public class StringToUpperConverter : IValueConverter
{
    public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
    {
        string val = value as string;
        return val != null ? val.ToUpper() : value;
    }

    public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
    {
        throw new NotImplementedException();
    }
}
```

# Comment utiliser les Bindings ?

- ConverterParameter : permet de passer un paramètre à votre Converter
- Utilisation des Converters

```
<StackLayout.Resources>
    <ResourceDictionary>
        <converters:StringToUpperConverter x:Key="StringToUpperConverter"/>
    </ResourceDictionary>
</StackLayout.Resources>
<Label Text="{Binding Name, Converter={StaticResource StringToUpperConverter}}"
    />
```

- Remarque : {StaticResource ResourceKey} vous permet d'accéder à ce que vous avez défini en resources
- /!\ Toutes les resources doivent avoir une clé ! (x:Key)

# Exercice 15

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 5312 Technologies Mobiles: Développement d'Applications Mobiles Cross-Platform avec Xamarin et C#

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

Xamarin - Les Web Services

# A propos de moi

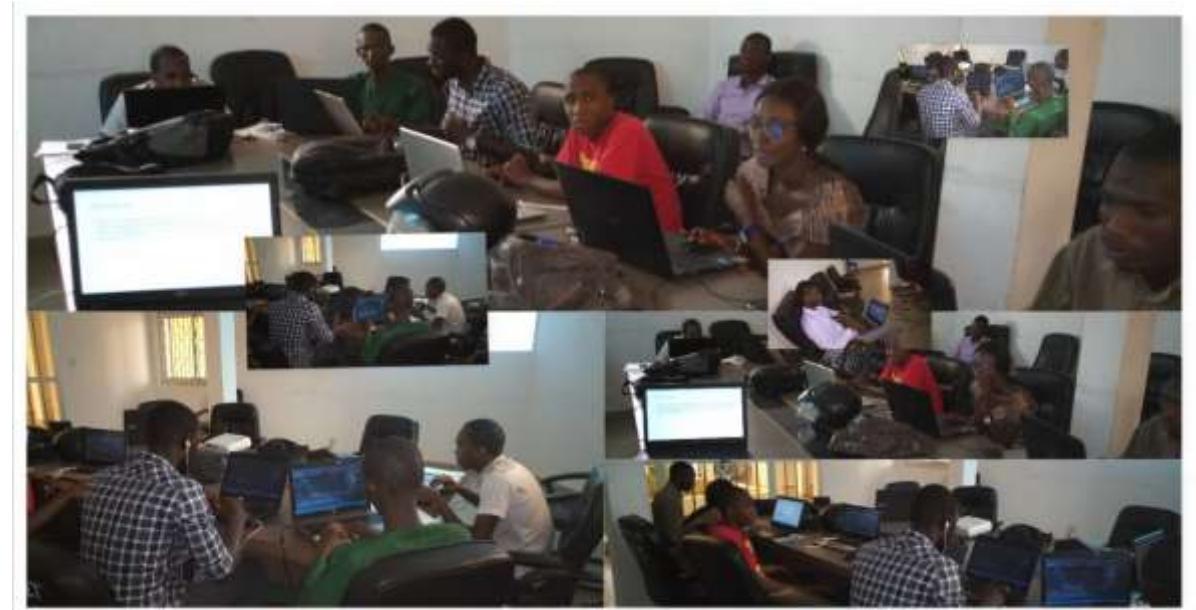


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Xamarin pour le développement d'Applications Mobiles
3. Une architecture pour le développement Cross-Plateforme
4. Développement Cross-Platform avec Xamarin.Forms
5. Interface utilisateur
6. Data-Binding
7. **Web Services**
8. Test et Déploiement

ExempleMySQLDB - Microsoft Visual Studio

Fichier Edition Affichage Projet Générer Déboguer Équipe Outils Test Analyser Fenêtre Aide

Debug Any CPU ExempleMySQLDB.Android AVD\_for\_Nexus\_One\_by\_Google (Android 7.1 - API 25)

NuGet - Solution App.xaml.cs

Parcourir Installé Mises à jour 9 Consolider

Microsoft.Net.Http   Inclure la version préliminaire

**.NET**

**Microsoft.Net.Http** par Microsoft, 18,6M téléchargements  
This package provides a programming interface for modern HTTP/REST based applications.

**Microsoft.Net.Http.zh-Hans** par Microsoft, 119K téléchargements  
Microsoft.Net.Http 程序包的 简体中文 资源

**Microsoft.Net.Http.es** par Microsoft, 28,4K téléchargements  
Recursos en español para el paquete Microsoft.Net.Http

**Microsoft.Net.Http.ja** par Microsoft, 19,3K téléchargements  
Microsoft.Net.Http パッケージの 日本語 リソース

**Microsoft.Net.Http.fr** par Microsoft, 19,9K téléchargements  
Ressources Français pour le package Microsoft.Net.Http

Sortie

Afficher la sortie à partir de : Gestionnaire de package

```
Tentative de collecte d'informations de dépendance pour le package 'Microsoft.Net.Http'.  
La collecte des informations de dépendance a pris 2,17 sec  
Tentative de résolution de dépendances pour le package 'Microsoft.Net.Http.2.2.29'.  
La résolution des informations de dépendance a pris 0 ms  
Actions en cours de résolution pour installer le package 'Microsoft.Net.Http.2.2.29'.  
Actions résolues pour installer le package 'Microsoft.Net.Http.2.2.29'.  
  
Tentative de collecte d'informations de dépendance pour le package 'Microsoft.Net.Http.2.2.29' du projet 'ExempleMySQLDB.iOS', ciblant 'Xamarin.iOS,Version=v1.0'  
La collecte des informations de dépendance a pris 1,62 sec  
Tentative de résolution de dépendances pour le package 'Microsoft.Net.Http.2.2.29' avec DependencyBehavior 'Lowest'  
La résolution des informations de dépendance a pris 0 ms  
Actions en cours de résolution pour installer le package 'Microsoft.Net.Http.2.2.29'.  
Actions résolues pour installer le package 'Microsoft.Net.Http.2.2.29'.
```

Gérer les packages de la solution

Source de package : nuget.org

ExempleMySQLDB.Android  ExempleMySQLDB.iOS

Acceptation de la licence

Vous devez accepter les termes des contrats de licence du ou des packages suivants avant de procéder à l'installation.

**Microsoft.Bcl** Auteur(s) : Microsoft

**Microsoft.Bcl.Build** Auteur(s) : Microsoft

**Microsoft.Net.Http** Auteur(s) : Microsoft

En cliquant sur "J'accepte", vous acceptez les termes du contrat de licence du ou des packages répertoriés ci-dessus. Si vous n'acceptez pas les termes du contrat de licence, cliquez sur "Je refuse".

Explorateur de solutions

Rechercher dans Explorateur de solution

Solution 'ExempleMySQLDB' (3 projets)

- ExempleMySQLDB (Portable)
- ExempleMySQLDB.Android
- ExempleMySQLDB.iOS

13:14 FRA 27/05/2017

ExempleMySQLDB - Microsoft Visual Studio

Fichier Edition Affichage Projet Générer Déboguer Équipe Outils Test Analyser Fenêtre Aide

Debug Any CPU ExempleMySQLDB.Android AVD\_for\_Nexus\_One\_by\_Google (Android 7.1 - API 25)

NuGet - Solution App.xaml.cs

Parcourir Installé Mises à jour Consolider

Newtonsoft.Json    Inclure la version préliminaire

Source de package : nuget.org

Newtonsoft.Json par James Newton-King, 59M téléchargements v10.0.2  
Json.NET is a popular high-performance JSON framework for .NET

NServiceBus.Newtonsoft.Json par NServiceBus Ltd, 20,9K téléchargements v1.1.0  
Newtonsoft.Json integration for NServiceBus.

CommonSerializer.Newtonsoft.Json par Brannon King, 4K téléchargements v1.0.1  
CommonSerializer.Newtonsoft Class Library

Newtonsoft.Json.FSharp par Henrik Feldt, Logibit AB, 17,6K téléchargements v3.2.2  
Different serializers for Newtonsoft.Json, making it easier to work with JSON data with Newtonsoft.Json from F#.

Sfa.Core.Newtonsoft.Json par Skills Funding Agency, 1,93K téléchargements v2.10.3  
Adds consistent patterns for working with Newtonsoft.Json when using the common Core Functionality.

Gérer les packages de la solution

Explorateur de solutions

Rechercher dans Explorateur de solution

Solution 'ExempleMySQLDB' (3 projets)  
ExempleMySQLDB (Portable)  
ExempleMySQLDB.Android  
ExempleMySQLDB.iOS

Sortie

Afficher la sortie à partir de : Gestionnaire de package

Tentative de collecte d'informations de dépendance pour le package 'Newtonsoft.Json.10.0.2' avec DependencyBehavior 'Lowest'  
La résolution des informations de dépendance a pris 0 ms  
Actions en cours de résolution pour installer le package 'Newtonsoft.Json.10.0.2'  
Actions résolues pour installer le package 'Newtonsoft.Json.10.0.2'

Tentative de collecte d'informations de dépendance pour le package 'Newtonsoft.Json.10.0.2' du projet 'ExempleMySQLDB', ciblant '.NETPortable,Version=v4.5,Profile=Profile259'  
La collecte des informations de dépendance a pris 723,49 ms  
Tentative de résolution de dépendances pour le package 'Newtonsoft.Json.10.0.2' avec DependencyBehavior 'Lowest'  
La résolution des informations de dépendance a pris 0 ms  
Actions en cours de résolution pour installer le package 'Newtonsoft.Json.10.0.2'  
Actions résolues pour installer le package 'Newtonsoft.Json.10.0.2'

Tentative de collecte d'informations de dépendance pour le package 'Newtonsoft.Json.10.0.2' du projet 'ExempleMySQLDB.Android', ciblant 'MonoAndroid,Version=v7.1'

Prêt Ln 1 Col 1 Car 1 INS Ajouter au contrôle de code source

ExempleMySQLDB - Microsoft Visual Studio

Fichier Edition Affichage Projet Générer Déboguer Équipe Outils Test Analyser Fenêtre Aide

Debug Any CPU ExempleMySQLDB.Android AVD\_for\_Nexus\_One\_by\_Google (Android 7.1 - API 25)

NuGet - Solution App.xaml.cs

Parcourir Installé Mises à jour 4 Consolider

Microsoft.BCL Inclure la version préliminaire

Source de package : nuget.org

Explorateur de solutions

Rechercher dans Explorateur de solution

Solution 'ExempleMySQLDB' (3 projets)

- ExempleMySQLDB (Portable)
- ExempleMySQLDB.Android
- ExempleMySQLDB.iOS

**Microsoft.Bcl** par Microsoft, 10M téléchargements v1.1.10

Adds support for types added in later versions of .NET when targeting previous versions.

**Acceptation de la licence**

Vous devez accepter les termes des contrats de licence du ou des packages suivants avant de procéder à l'installation.

**Microsoft.Bcl.Build** Auteur(s) : Microsoft  
[Afficher la licence](#)

**Microsoft.Bcl** Auteur(s) : Microsoft  
[Afficher la licence](#)

En cliquant sur "J'accepte", vous acceptez les termes du contrat de licence du ou des packages répertoriés ci-dessus. Si vous n'acceptez pas les termes du contrat de licence, cliquez sur "Je refuse".

J'accepte Je refuse

.NET Microsoft.Bcl

Version(s) - 0

Projet ^	Version
<input checked="" type="checkbox"/> ExempleMySQLDB	
<input checked="" type="checkbox"/> ExempleMySQLDB.Android	
<input type="checkbox"/> ExempleMySQLDB.iOS	

Installé : non installé Désinstaller

Version : Dernière version stable 1.1.10 Installer

Tentative de collecte d'informations de dépendance pour le package 'Microsoft.Bcl.1.1.10'. La collecte des informations de dépendance a pris 1,76 sec. Tentative de résolution de dépendances pour le package 'Microsoft.Bcl.1.1.10'. La résolution des informations de dépendance a pris 0 ms. Actions en cours de résolution pour installer le package 'Microsoft.Bcl.1.1.10'. Actions résolues pour installer le package 'Microsoft.Bcl.1.1.10'.

Tentative de collecte d'informations de dépendance pour le package 'Microsoft.Bcl.1.1.10' du projet 'ExempleMySQLDB.Android', ciblant 'MonoAndroid, Version=v7.1'. La collecte des informations de dépendance a pris 503,33 ms. Tentative de résolution de dépendances pour le package 'Microsoft.Bcl.1.1.10' avec DependencyBehavior 'Lowest'. La résolution des informations de dépendance a pris 0 ms. Actions en cours de résolution pour installer le package 'Microsoft.Bcl.1.1.10'. Actions résolues pour installer le package 'Microsoft.Bcl.1.1.10'.

Liste d'erreurs Sortie

Prêt Ajouter au contrôle de code source

# Exercice 16

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# **INF 5312 Technologies Mobiles:**

## **Développement d'Applications**

## **Mobiles Cross-Platform avec Xamarin**

## **et C#**

Master Informatique Option Génie Logiciel

Année Universitaire 2016-2017

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

**Partie 8:**  
**Tests et Déploiements**

# A propos de moi

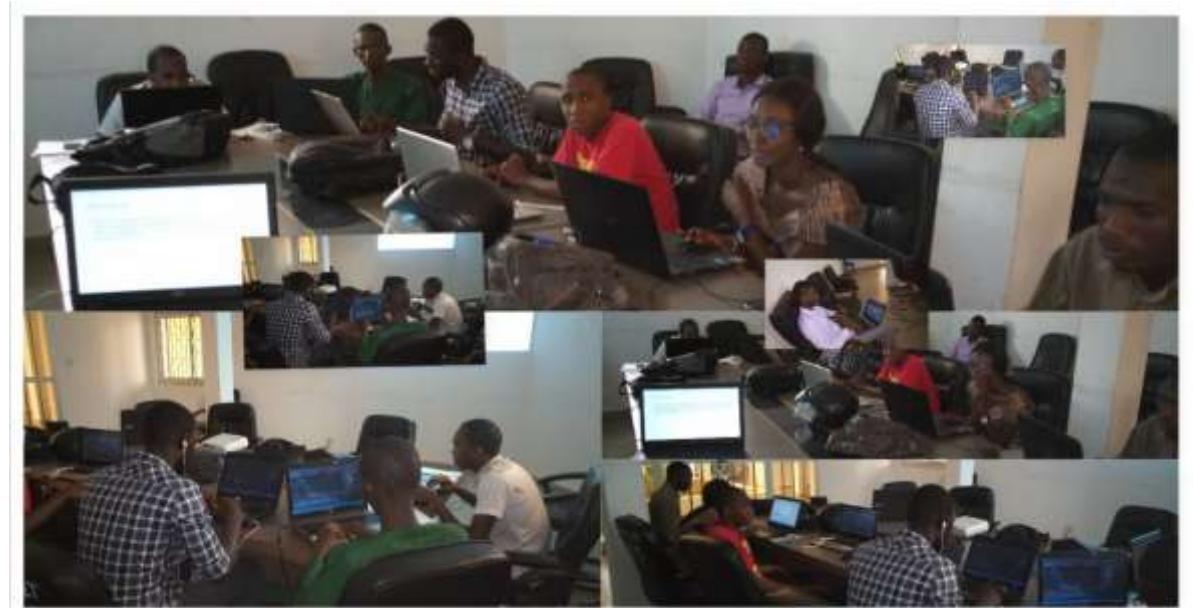


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE
  - Technologies Mobiles Android, Xamarin
  - Programmation .Net, C#
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Xamarin pour le développement d'Applications Mobiles
3. Une architecture pour le développement Cross-Plateforme
4. Développement Cross-Platform avec Xamarin.Forms
5. Interface utilisateur
6. Data-Binding
7. Web Services
- 8. Test et Déploiement**

# Exercice 17

# Webography

- <https://developer.xamarin.com/guides/>
- <https://docs.microsoft.com/en-us/aspnet/core/>
- <http://scr.sad.supinfo.com/articles/single/3532-construire-une-interface-utilisateur-avec-xamarin#idm140445263718048>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.



TypeScript



# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles hybrides



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique



**Partie 5: Développement d'Applications Mobiles Hybrides  
avec Cordova et le framework Ionic 4**



# A propos de moi

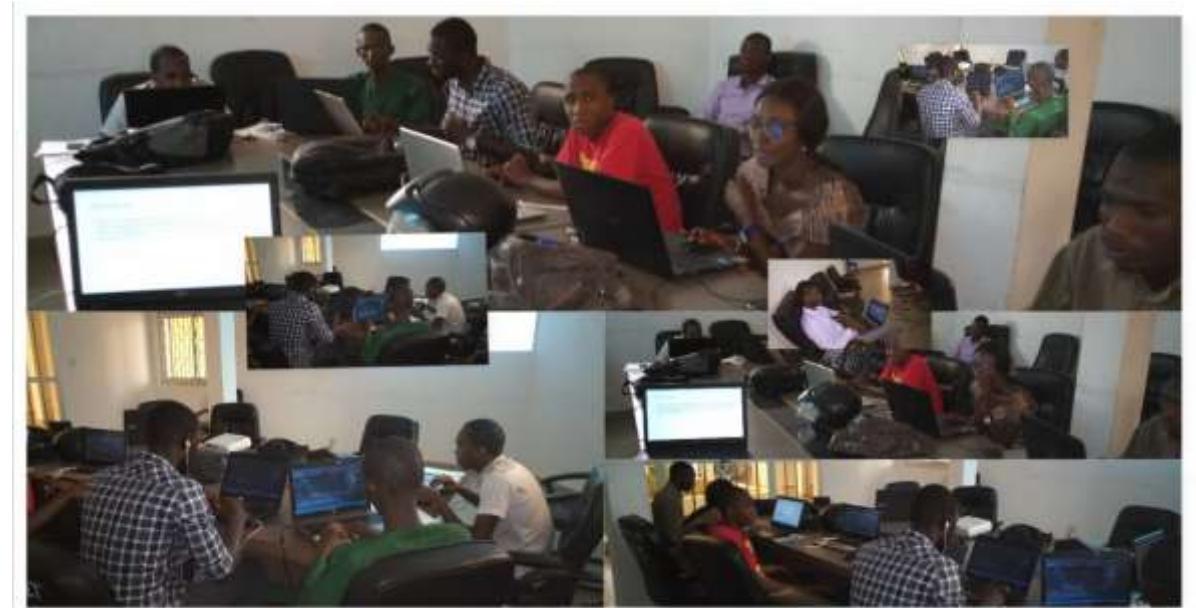


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
3. Approches de développement mobile
4. Xamarin pour le développement d'Applications Mobiles
5. **Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4**
6. Développement d'Applications Natives avec Android



# Ionic

*Ionic* est un framework de développement qui permet de créer des applications hybrides en HTML5, CSS, Javascript. Il est basé sur des frameworks/technologies qui ont fait leurs preuves telles que Angular et Apache Cordova.

**Apache Cordova** < Application

Apache Cordova ou plus anciennement Apache Callback ou PhoneGap, est un framework open-source développé par la Fondation Apache. Il permet de créer des applications pour différentes plateformes en HTML, CSS et JavaScript. [Wikipédia](#)

**Licence :** [Licence Apache v2 \(Cordova License\)](#)

**Développé par :** [Fondation Apache](#)

**Dernière version :** 8.1.1 (27 septembre 2018)

**Système d'exploitation :** Android

**Type :** Framework de développement hybride pour smartphone (en)

**Programmé en :** C#, C++, Feuilles de style en cascade, Hypertext Markup Language, Java, JavaScript, Objective-C

**Recherches associées** [Voir d'autres éléments \(plus de 15\)](#)

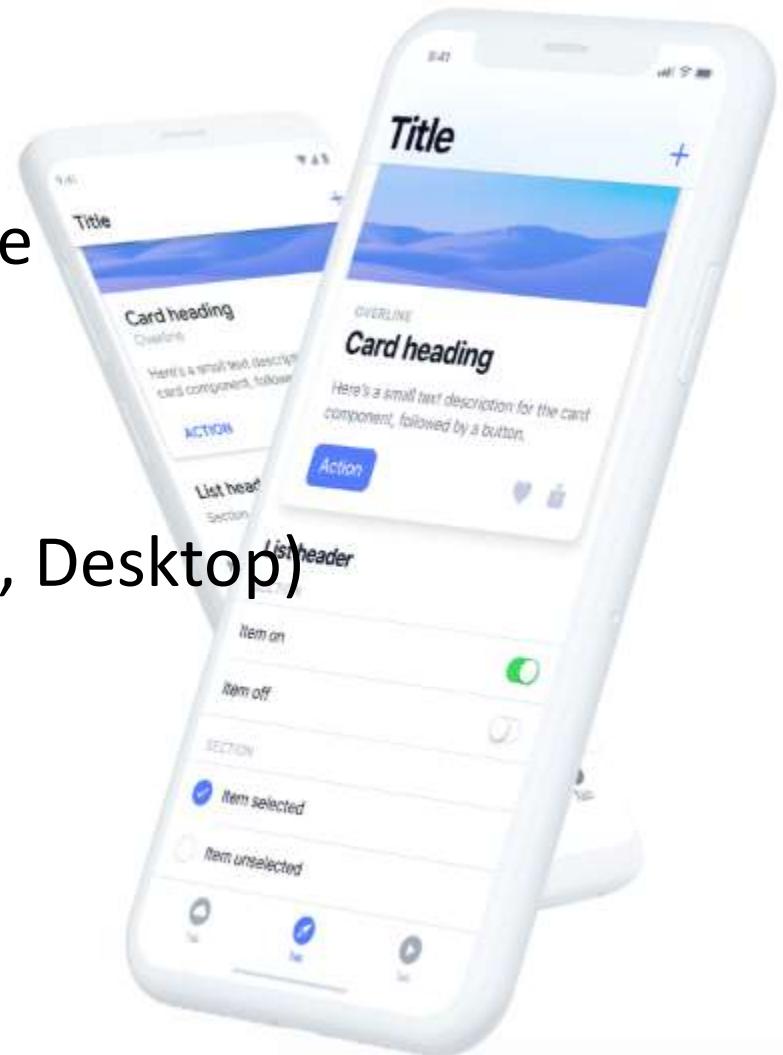
 Ionic  PhoneGap  AngularJS  Node.js  npm



<b>Developer(s)</b>	<a href="#">Drifty</a> <sup>[1][2]</sup>
<b>Initial release</b>	2013; 6 years ago
<b>Stable release</b>	4.1.1 <sup>[3]</sup> / 7 March 2019; 10 days ago
<b>Repository</b>	<a href="#">github.com/ionic-team/ionic</a> 
<b>Written in</b>	<a href="#">JavaScript</a>
<b>Type</b>	<a href="#">Software framework</a>
<b>License</b>	<a href="#">MIT License</a>
<b>Website</b>	<a href="#">ionicframework.com</a> 

# IONIC

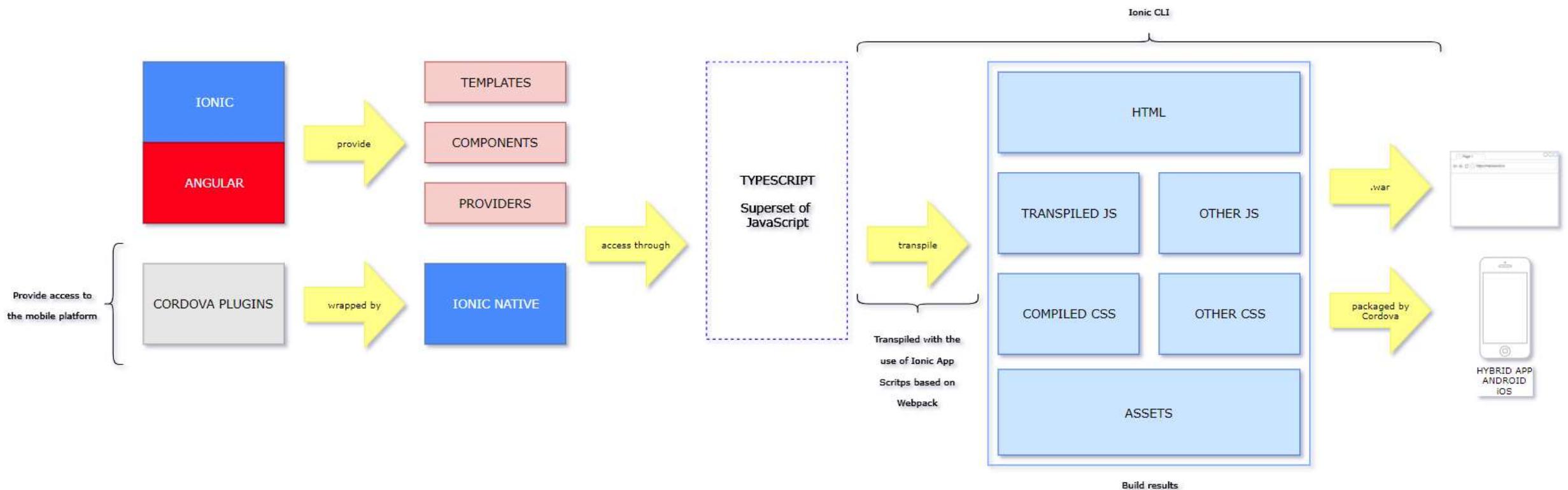
- Boîte à outils d'interface utilisateur Open Source
- Crée et maintenu par l'équipe ionique
- Construit à l'aide de pochoir
- Développement multiplateforme (PWA, Mobile, Desktop)



# IONIC 1, 2, 3 et 4

- IONIC 1 est basé sur Angular 1 (Angular JS) :
  - Première version de IONIC qui est la plus populaire.
  - Angular JS est basé sur une architecture MVC coté client. Les applications IONIC 1 sont écrite en Java Script.
- IONIC 2 est basé Angular 2 (Angular) :
  - Angular 2 est une réécriture de Angular 1 qui est plus performante, mieux structurée et représente le futur de Angular.
  - Les applications de Angular2 sont écrire en Type Script qui est compilé et traduit en Java Script avant d'être exécuté par les Browsers Web.
  - Angular 2 est basée sur une programmation basée sur les Composants Web (Web Components)
- IONIC 3 est basé sur Angular 4 est une simple mise à jour de IONIC 2  
(La version Angular 3 a été abandonnée)
- IONIC 4 depuis Novembre 2018

# Schéma explicatif de IONIC



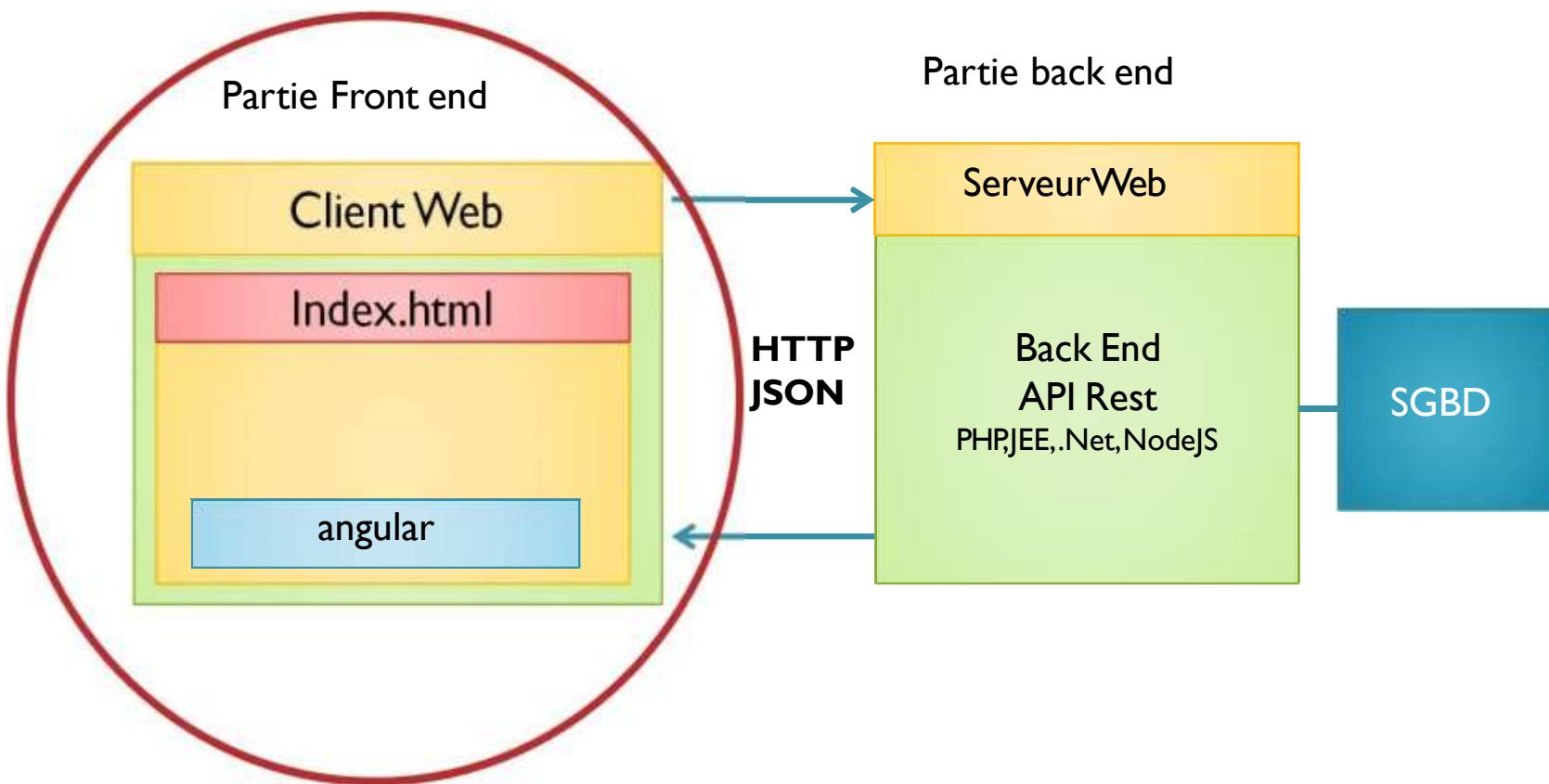
# Angular: Qu'est-ce que Angular

- Framework front end JavaScript
- Crée et maintenu par Google
- Différent d'AngularJS
- Utilisé pour construire des applications frontend puissantes
- Développement rapide et génération de codes

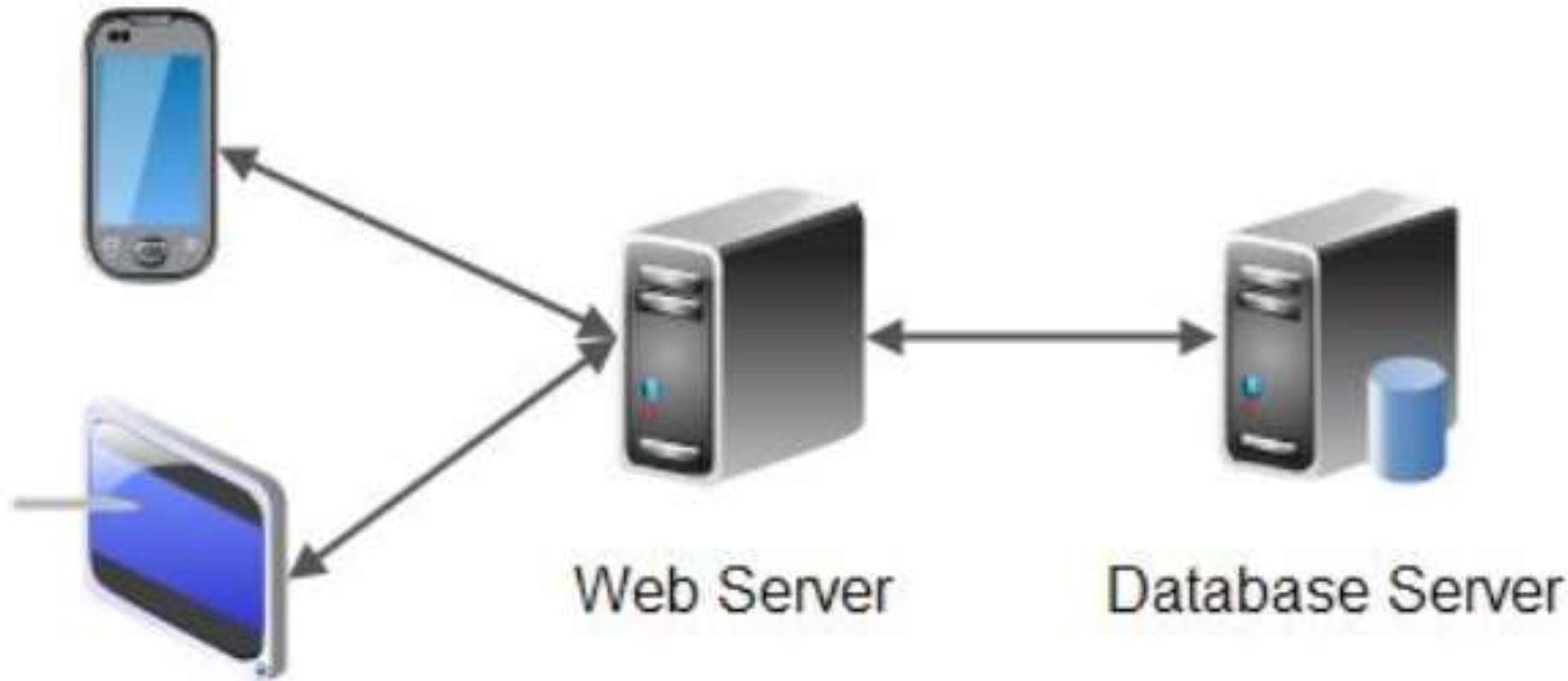
# Angular

- Angular Permet de créer la partie front end des applications web de type SPA (Single Page Application réactive)
- Une SPA est une application qui contient une seule page HTML (index.html) récupérée du serveur.
- Pour naviguer entre les différentes parties de cette application, Java Script est utilisé pour envoyer des requêtes http (AJAX) vers le serveur pour récupérer du contenu dynamique généralement au format JSON.
- Ce contenu JSON est ensuite affiché côté client au format HTML dans la même page.

# Angular

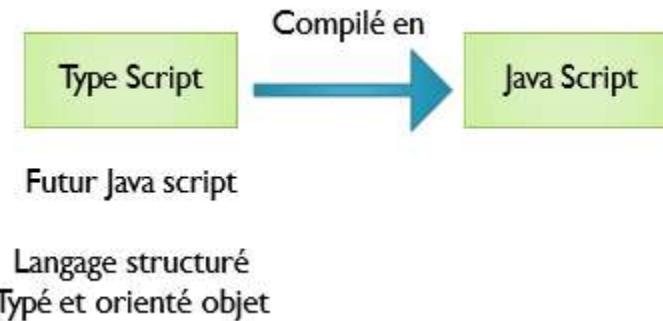


# Architecture 3-tiers



# Angular avec Type Script

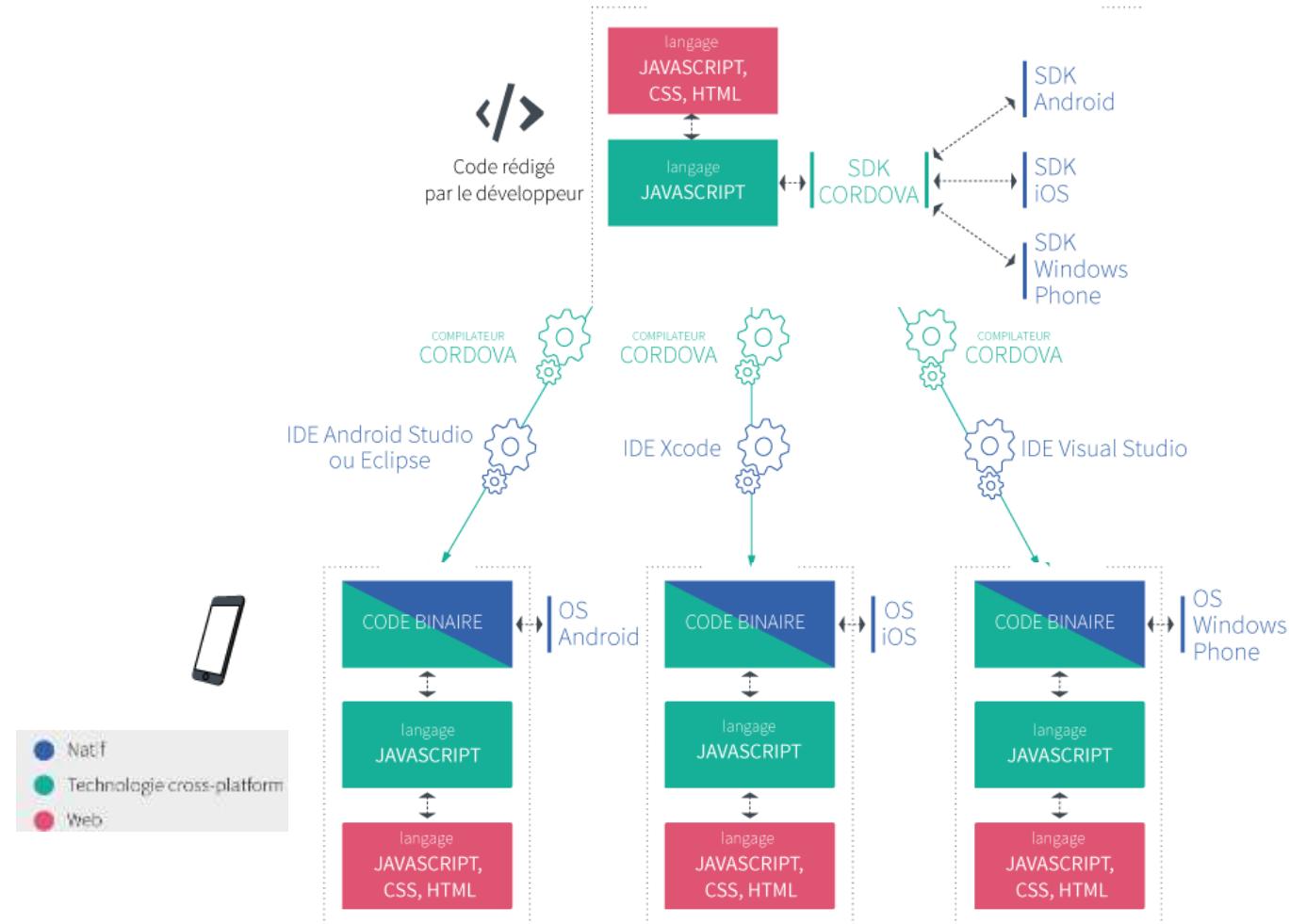
- Pour développer une application Angular il est recommandé d'utiliser Type Script qui sera compilé et traduit en Java Script.
- Type Script est un langage de script structuré et orienté objet qui permet de simplifier le développement d'applications Java Script



# Rappel: Application mobile hybride

- Une application hybride combine
  - Des éléments d'une application web HTML5, CSS, Java Script
    - Pour la partie Présentation des éléments de l'interface on utilise
      - Des Framework CSS créées pour les applications mobiles (Boot Strap)
      - Des Framework Java Script ( Angular JS, JQuery Mobile, Dojo, Sencha Touch)
    - Pour la partie Interaction avec le serveur, on utilise des Frameworks Ajax ( Angular, JQuery Mobile, ...)
  - Des éléments d'une application native permettent d'utiliser toutes les fonctionnalités natives des smartphones ((GPS, Local Storage, Push Notifications, SMS, Appels Téléphoniques, Caméra, Accéléromètre, Mode connecté et mode non connecté, Gestion de l'énergie, Mémoires embarquées, etc.)
    - Pour utiliser interagir avec les fonctionnalités natives, l'application mobile native utilise les plugins du Framework CORDOVA (Apache) ou PhoneGap (Adobe).
    - En plus Cordova offre des outils qui permettent de :
      - Générer des applications mobiles natives pour les différentes plateformes
      - Tester ces applications sur des émulateurs appropriés
  - De plus elle pourra être distribuée en tant qu'application sur les plateformes d'applications (App Store, Google Play Store, Windows Store)

# Développement Mobile Hybride



# Plateforme de développement mobile hybride

- Appcelerator Titanium (gratuit)
- Win Dév mobile (payant)
- IBM Worklight (MobileFiorst) (gratuit dans sa version développeur et payant la partie serveur)
- Adobe Phone Gap (Payant à partir d'une application de plus de 50 MB )
- IONIC (GRATUIT, OPENSOURCE)

# Qu'est-ce que IONIC Framework ?

- Ionic Framework est un mélange d'outils et de technologies pour développer des applications mobiles hybrides rapidement et facilement.
- Il s'appuie sur :
  - Angular pour la partie application web du framework
  - Cordova pour la partie construction des applications natives.
- Il s'appuie aussi sur la plateforme NodeJS :
  - Plus précisément NPM (Node Package Manager)
  - Et pour démarrer un serveur web local basé sur NodeJS pour tester la partie web de l'application en local.
- Ce Framework open source permet de développer une application déployable sur plusieurs environnements tel que :
  - Application mobile Web
  - Une application mobile pour des systèmes tel que : Android, iOS, Windows Phone...

# Installation

- Installer d'abord Node.js
- Ensuite, installer Cordova et Ionic.
- Pour les applications Android, il faut installer le SDK de Android
- Pour les application IOS, il faut installer XCode.

# Installer et configurer Cordova

1. Installer NodeJS (<https://nodejs.org/>). NodeJS est une plateforme événementielle en JavaScript orientée vers les applications réseaux avec une bibliothèque de serveur HTTP intégrée.
2. Installer Ionic et Cordova à partir de l'invité de commandes ( cmd - admin):  
`>npm install -g ionic cordova`
3. Installer la dernière version du Java Development Kit à partir du site  
`>http://www.oracle.com/technetwork/java/javase/downloads/`  
Déclarer la variable d'environnement JAVA\_HOME pointant vers la racine du dossier Java, Ex: C:\Program Files\Java\jdk1.8.0\_65.
4. Installer le SDK Android (<http://developer.android.com/sdk/>)  
Ajouter les dossiers tools et platform-tools dans le PATH des variables d'environnement. Sinon installer Android Studio simplement  
<https://developer.android.com/studio/>.
5. Si absent, installer Ant (<http://ant.apache.org/bindownload.cgi>) et l'ajouter au PATH du système. Ant gère la compilation du projet.

# Capacitor au lieu de Cordova

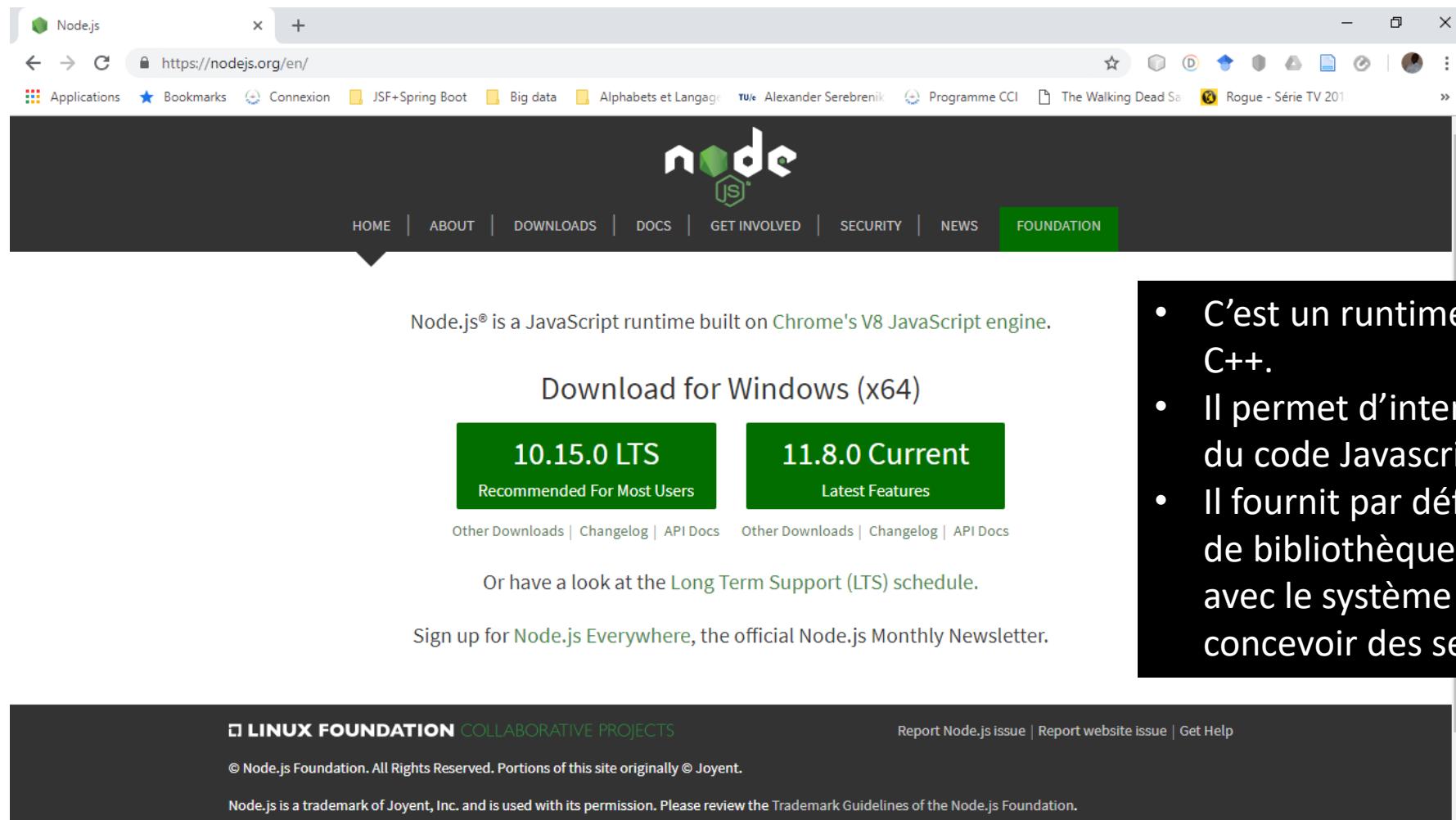
## The Native Bridge for Cross-Platform Web Apps

Invoke Native SDKs on iOS, Android, Electron, and the Web with one code base. Optimized for Ionic Framework apps, or use with any web app framework.

[Get Started](#)

SUPPORTS

# Installer NodeJS ( <https://nodejs.org/> )



The screenshot shows a web browser window with the Node.js homepage loaded. The URL in the address bar is <https://nodejs.org/en/>. The browser's toolbar includes icons for Applications, Bookmarks, Connexion, JSF+Spring Boot, Big data, Alphabets et Langage, Alexander Serebrenik, Programme CCI, The Walking Dead S, Rogue - Série TV 201, and more.

The Node.js website features a dark header with the Node.js logo and navigation links for HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, NEWS, and FOUNDATION. The FOUNDATION link is highlighted with a green background.

The main content area starts with a paragraph: "Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine." Below this is a large button labeled "Download for Windows (x64)". Underneath the button are two green boxes: one for "10.15.0 LTS" (Recommended For Most Users) and another for "11.8.0 Current" (Latest Features). Below these boxes are links for "Other Downloads", "Changelog", and "API Docs".

Further down the page, there is a link to "Or have a look at the Long Term Support (LTS) schedule." and a call to action: "Sign up for Node.js Everywhere, the official Node.js Monthly Newsletter."

The footer of the page includes the Linux Foundation logo and the text "COLLABORATIVE PROJECTS". It also contains links for "Report Node.js issue", "Report website issue", and "Get Help". The footer also states: "© Node.js Foundation. All Rights Reserved. Portions of this site originally © Joyent." and "Node.js is a trademark of Joyent, Inc. and is used with its permission. Please review the Trademark Guidelines of the Node.js Foundation."

**• C'est un runtime Javascript écrit en C++.**  
**• Il permet d'interpréter et exécuter du code Javascript.**  
**• Il fournit par défaut un ensemble de bibliothèques pour interagir avec le système d'exploitation et concevoir des serveurs Web.**

# NPM: Node Package Manager

- Lorsque vous installez node, vous installez également npm.
- C'est un outil en ligne de commande qui vous permet de facilement installer des packages écrits en Javascript et compatibles avec NodeJS.
- Pour rechercher des packages rendez-vous sur <https://npmjs.com>





# Installer Ionic et Appache Cordova

- Les structures des applications de IONIC sont créées grâce à des commandes IONIC (Ionic CLI). Pour installer d'une manière globale IONIC et Cordova, exécuter la commande :

```
C:\Users\OSall>npm install -g cordova ionic
```

Stack Overflow

Site web

Stack Overflow est un site web proposant des questions et réponses sur un large choix de thèmes concernant la programmation informatique. Il fait partie du réseau de sites Stack Exchange. [Wikipedia](#)

Inscription : Facultative

Créé par : Joel Spolsky et Jeff Atwood

Lancement : Août 2008

Date de lancement : 15 septembre 2008

Créateur : Joël Spolsky, Jeff Atwood

Si vous rencontrez des messages d'erreurs copier et coller les sur votre navigateur pour voir les solutions proposées. Particulièrement sur le site de StackOverflow  
<https://stackoverflow.com/>

C:\Users\OSall>ionic --v



Usage:

```
$ ionic <command> [<args>] [--help] [--verbose] [--quiet] [--no-interactive] [--no-color] [--confirm] [options]
```

Global Commands:

config <subcommand>	.....	Manage CLI and project config values (subcommands: <code>get</code> , <code>set</code> , <code>unset</code> )
docs	.....	Open the Ionic documentation website
info	.....	Print project, system, and environment information
init	.....	(beta) Initialize existing projects with Ionic
login	.....	Login to Ionic Appflow
logout	.....	Logout of Ionic Appflow
signup	.....	Create an account for Ionic Appflow
ssh <subcommand>	.....	Commands for configuring SSH keys (subcommands: <code>add</code> , <code>delete</code> , <code>generate</code> , <code>list</code> , <code>setup</code> , <code>use</code> )
start	.....	Create a new project

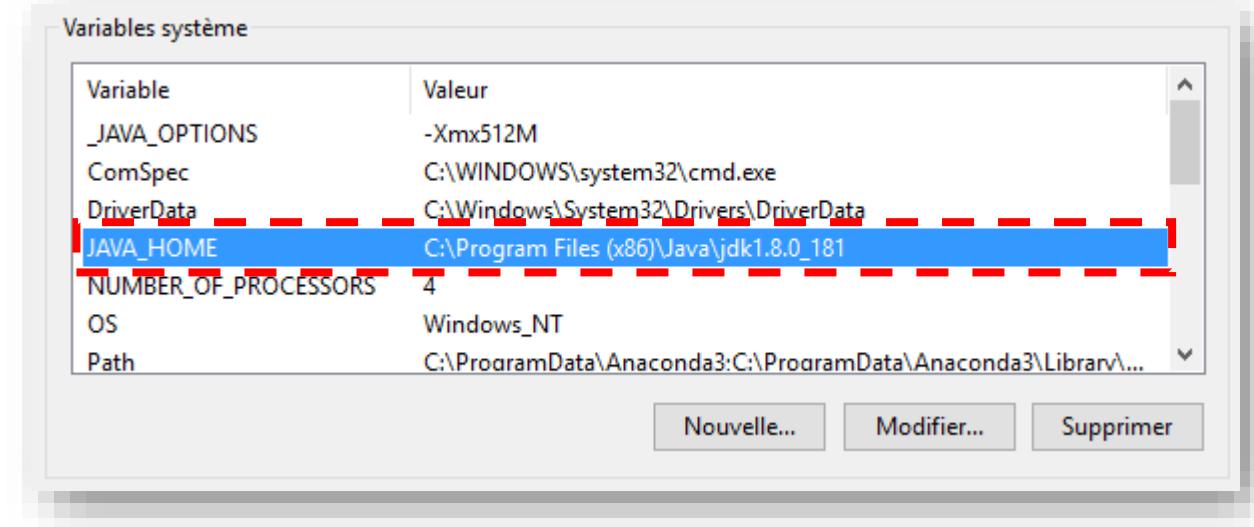
# Cordova

Cordova est un framework qui permet de packager et distribuer un application web sur des périphériques mobiles

- Apache Cordova est une plateforme pour construire des applications natives mobiles en utilisant HTML5, CSS et Java Script
- Suite au rachat de PhoneGap par Adobe, l'ensemble du SDK cross- platform PhoneGap a été rebaptisé Cordova et a été introduit dans l'incubateur de la fondation Apache.
- Cordova représente aujourd'hui la meilleure solution de développement cross- platflorm du marché.
- Elle permet avec peu d'efforts de développer une application mobile une fois et de la faire fonctionner sur toutes les plateformes mobiles du marché.
- Les avantages de Cordova sont nombreux :
  - Cordova est OpenSource (Licence Apache)
  - Cordova est basée sur les standards du Web.
  - Cordova supporte la plupart des plateformes mobiles du marché (Android, iOS, Blackberry, Windows Phone 7 ...).

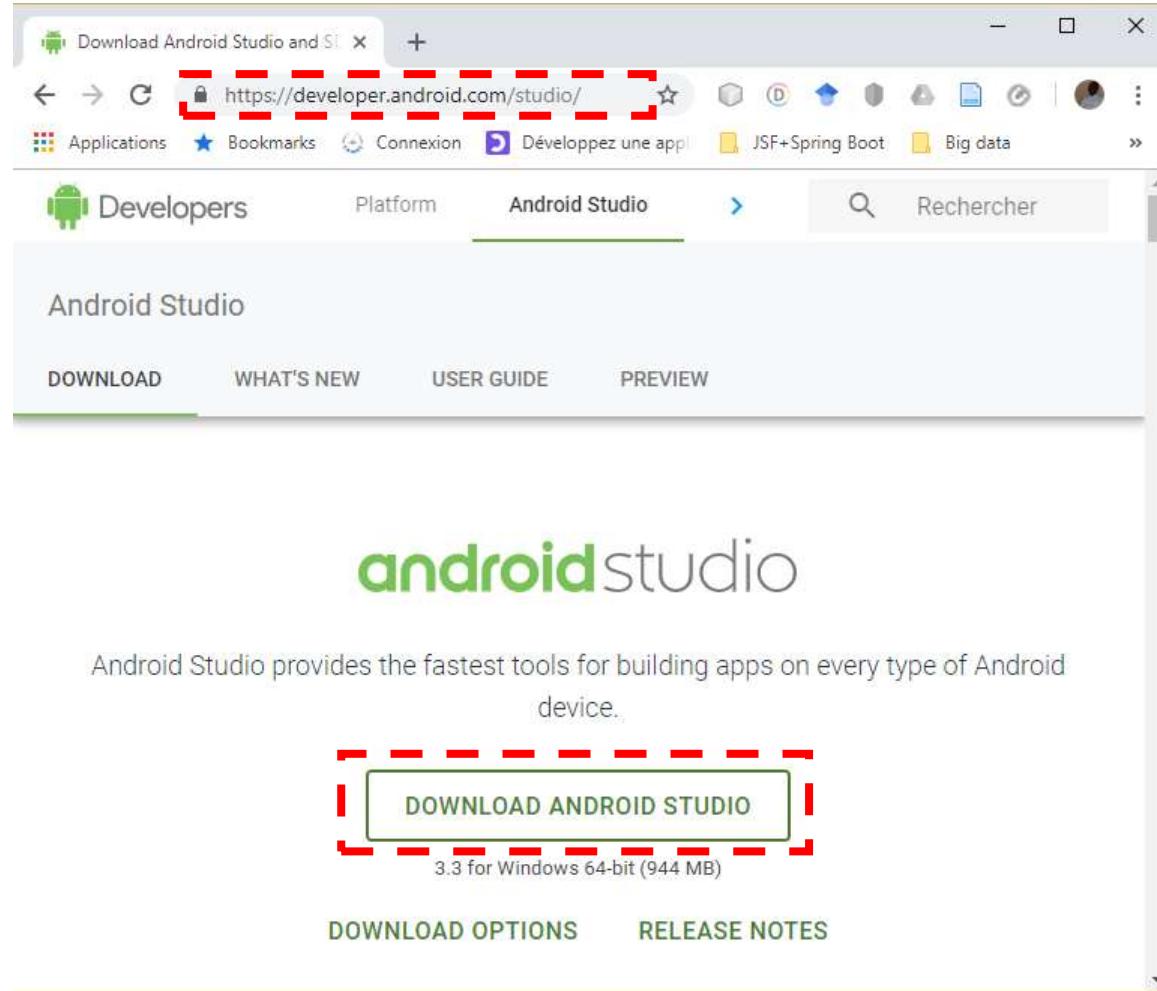
# Installer la dernière version du Java Développement Kit à partir du site

- <http://www.oracle.com/technetwork/java/javase/downloads/>
- Déclarer la variable d'environnement JAVA\_HOME pointant vers la racine du dossier Java, Ex: C:\Program Files\Java\jdk1.8.0\_65



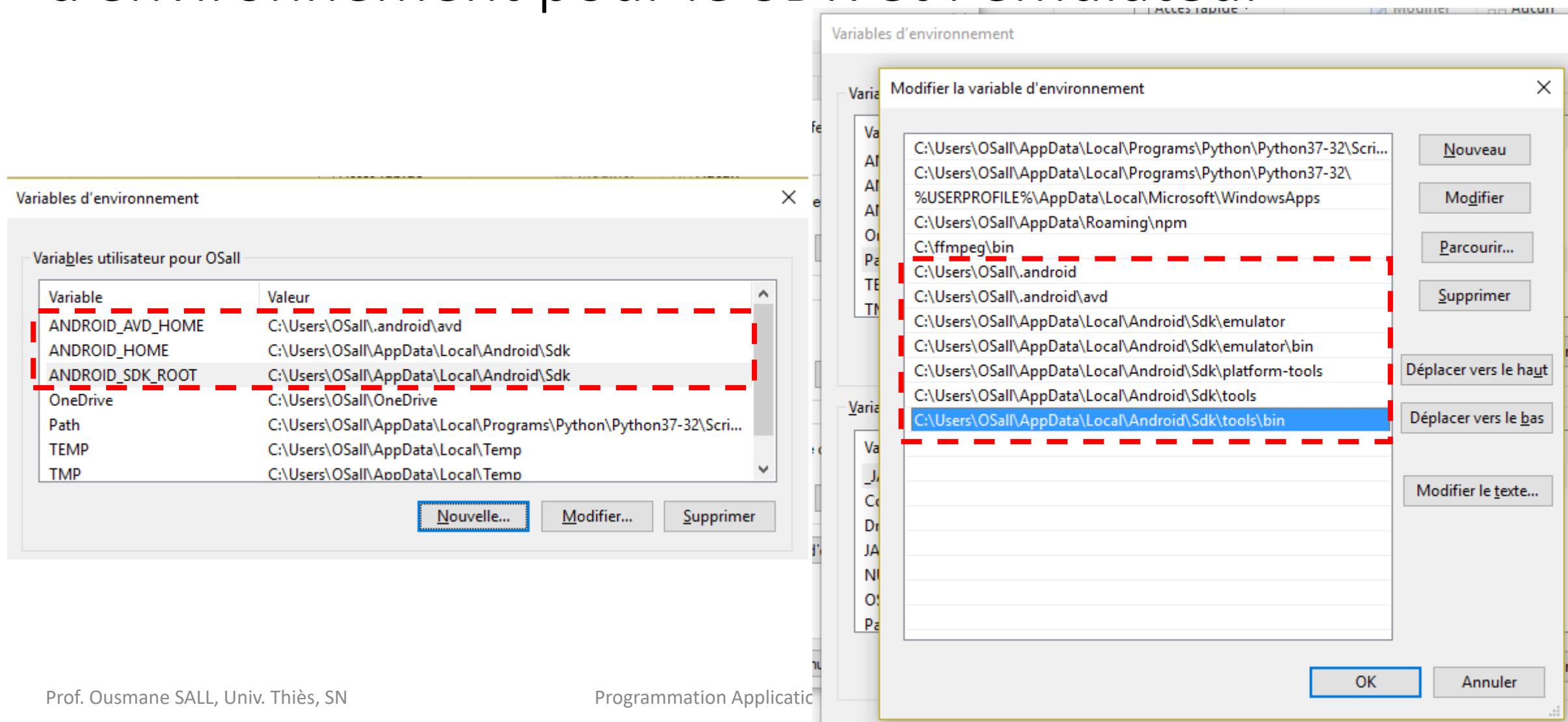
# Installer Android Studio

<https://developer.android.com/studio/>



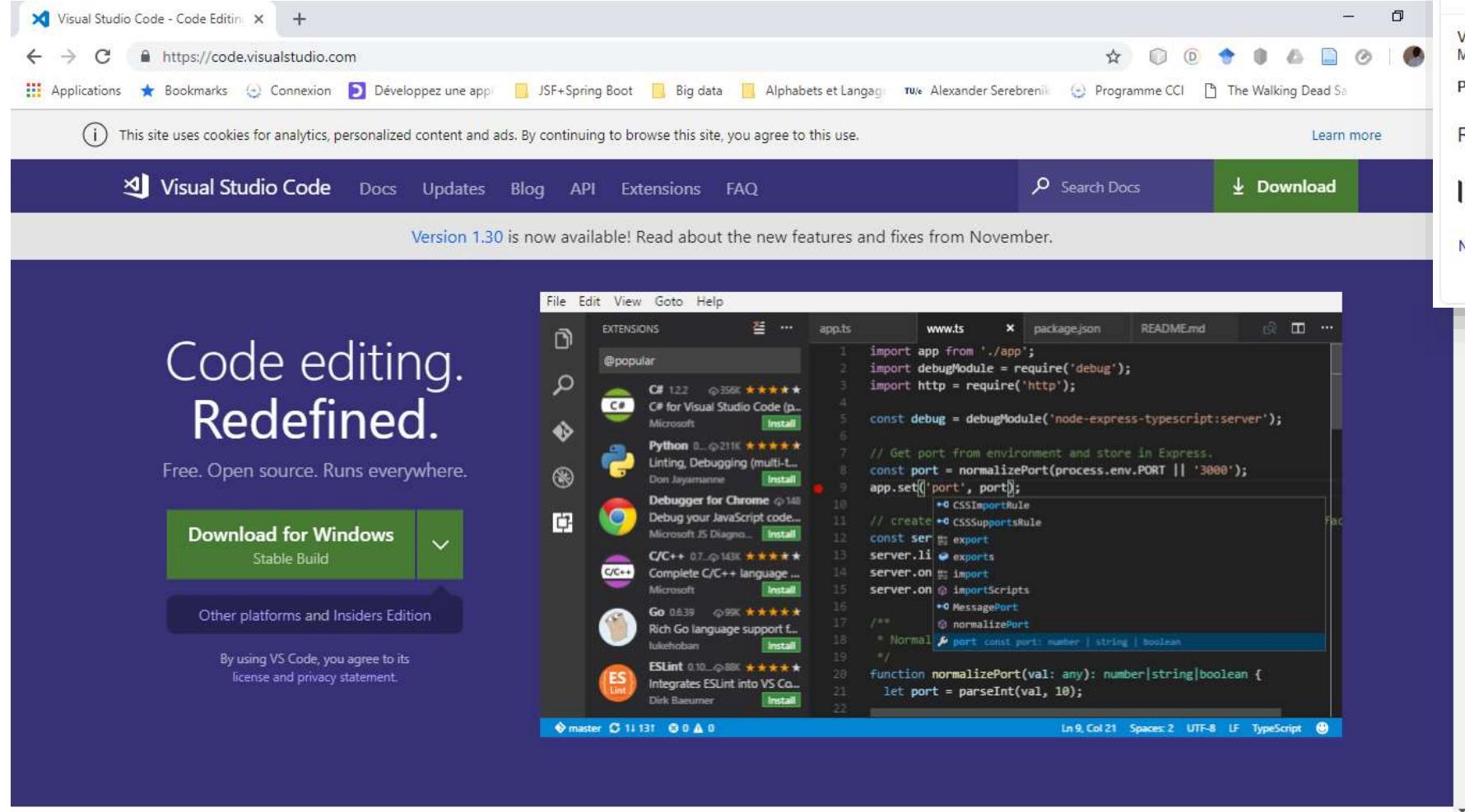
Vous devez au moins créer un émulateur après l'installation sous Android Studio

# Installer Android Studio: Configurer les variables d'environnement pour le SDK et l'émulateur



# Installer Visual Studio Code

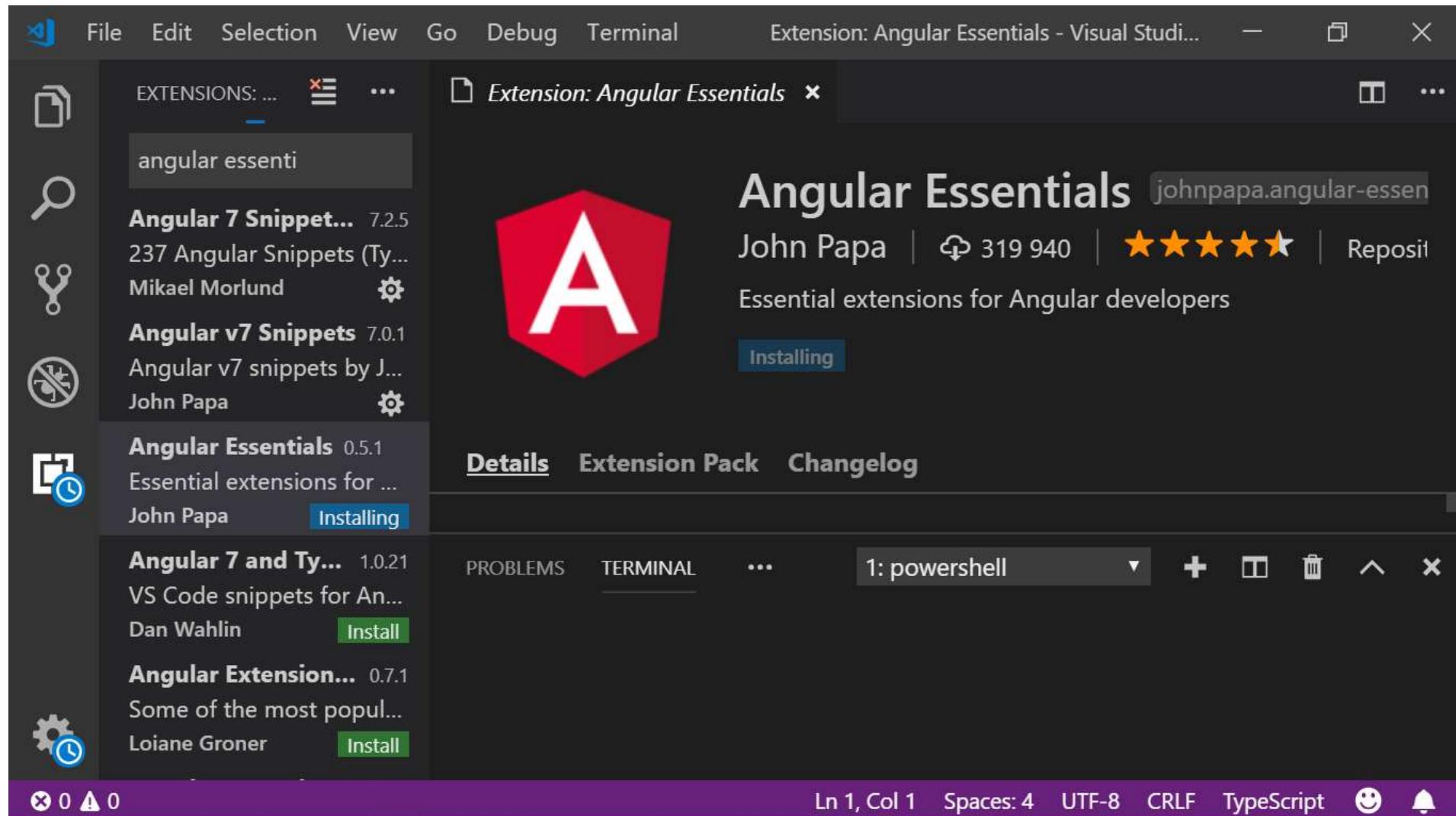
## <https://code.visualstudio.com/>



The screenshot shows the official Visual Studio Code website at <https://code.visualstudio.com/>. The page features a large banner with the text "Code editing. Redefined." and "Free. Open source. Runs everywhere." Below the banner are download buttons for "Download for Windows" (Stable Build) and "Other platforms and Insiders Edition". A note states: "By using VS Code, you agree to its license and privacy statement." The main content area displays the Visual Studio Code interface with a code editor showing TypeScript code and a sidebar with various extensions like C#, Python, and Debugger for Chrome.



# Installer Visual Studio Code: extensions



# Installer Visual Studio Code: extensions

The screenshot shows the Visual Studio Code interface with the Extensions view open. The search bar at the top left contains the text "material icon theme". The main panel displays the "Material Icon Theme" extension by pkief, version 3.6.3. The extension's icon is a blue folder with a white icon inside. The title "Material Icon Theme" is followed by the author's name "Philipp Kief", the download count "6 005 065", a five-star rating, and links to "Repository" and "License". Below the title, the description reads "Material Design Icons for Visual Studio Code". At the bottom of the extension card are three buttons: "Set File Icon Theme", "Disable", and "Uninstall".

**Details** Contributions Changelog

Azure-pipelines	Eslint	Jupyter	Python-misc	TypeScript-def
Babel	Exe	Karma	R	Url
Ballerina	Fastlane	Key	Racket	Vagrant
Bazel	Favicon	Kivy	Raml	Velocity
Bitbucket	Firebase	KI	Razor	Verilog
Bithound	Flash	Kotlin	React	Vfl
Blink	Flow	Laravel	React_ts	Video
Bower	Font	Less	Readme	Virtual

File Edit Selection View Go Debug Terminal Help Extension: Material Icon Theme - Visual Studio Code

EXTENSIONS: ... ⚙️ ...

material icon theme

**Material Icon The...** 3.6.3

Material Design Icons fo... Philipp Kief

**Angular 7 Snippet...** 7.2.5

237 Angular Snippets (Ty... Mikael Morlund

**vscode-icons** 8.3.0

Icons for Visual Studio C... VSCode Icons Team

**Material Theme** 2.8.0

The most epic theme no... Mattia Astorino **Install**

**Nomo Dark Icon T...** 1.3.6

Nomo Dark Icon Theme

**be5invis** **Install**

**Deepdark Materi...** 2.5.2

Clean real dark material ...

**Nimda** **Install**

**Angular Material ...** 0.13.0

Providers snippets Anaul...

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF TypeScript

49

# Installer Visual Studio Code: extensions

The screenshot shows the Visual Studio Code interface with the Marketplace extension installed. The left sidebar lists various extensions, and the main panel displays the details for the 'ionic4-snippets-vscode' extension by Arkade.

**Extension: ionic4-snippets-vscode - exemplien - Visual Studio Code**

**ionic4-snippets-vscode** by Arkade | 5028 | ★★★★★ | Repository

ionic 4 code snippets

**Details Contributions Changelog**

You just need to backspace out of the array selection and then tab into the next part of the snippet

\*\* I am not sure how to get around that issue - this issue is open here

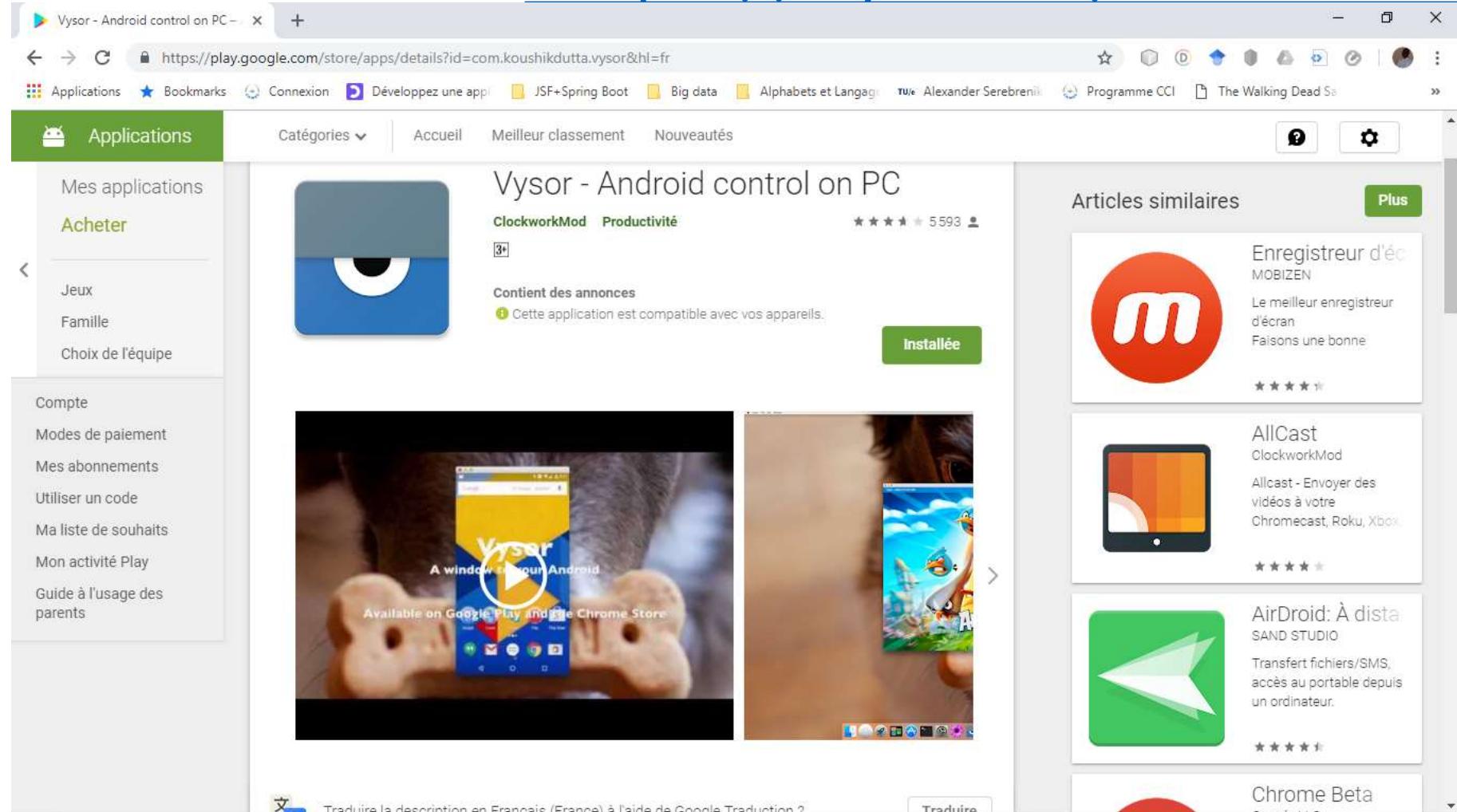
A preview of the snippets is shown in the code editor:

```
i4-
<ion-c>
  <i4-button-icon-left>
  <i4-avatar>
  <i4-backdrop>
  <i4-backdrop-dynamic>
  <i4-badge slot="start">
  <i4-button>
```

Shortcut Code	Component	Options
i4-avatar	ion-avatar with nested img tag	slot
i4-button-back	nested ion-back-button in ion-buttons tag	defaultHref

Bottom status bar: Ln 27, Col 1 Spaces: 2 UTF-8 LF HTML

# Installer Vysor sur votre téléphone et sur votre ordinateur <https://vysor.io/download/>



# Création du projet Ionic

- Lors de la création d'applications dans Ionic, vous pouvez choisir parmi les trois options suivantes pour spécifier le type de template du projet (Voir [dapo](#) suivante pour les autres types de template):
  - **Tabs App**
  - **Blank App**
  - **Sidemenu App**
- Dans la fenêtre de commande, ouvrez le dossier dans lequel vous souhaitez créer l'application et essayez l'une des options mentionnées ci-dessous.
- Commande: **ionic start nomApplication --type=angular**

# Création du projet: Templates

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [version 10.0.17134.523]
(c) 2018 Microsoft Corporation. Tous droits réservés.

C:\Users\OSall>ionic start --list
name      | project type | description
-----
blank     | angular      | A blank starter project
sidemenu  | angular      | A starting project with a side menu with navigation in the content area
tabs      | angular      | A starting project with a simple tabbed interface
tabs      | ionic-angular | A starting project with a simple tabbed interface
blank     | ionic-angular | A blank starter project
sidemenu  | ionic-angular | A starting project with a side menu with navigation in the content area
super    | ionic-angular | A starting project complete with pre-built pages, providers and best practices for Ionic development.
tutorial  | ionic-angular | A tutorial based project that goes along with the Ionic documentation
aws       | ionic-angular | AWS Mobile Hub Starter
tabs      | ionic1       | A starting project for Ionic using a simple tabbed interface
blank     | ionic1       | A blank starter project for Ionic
sidemenu  | ionic1       | A starting project for Ionic using a side menu with navigation in the content area
maps     | ionic1       | An Ionic starter project using Google Maps and a side menu

C:\Users\OSall>
```

# Création d'un projet Ionic 4

```
C:\Users\OSall\Desktop\AppIonic>ionic start tabs --type=angular
```

Let's pick the perfect starter template!

Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your app. To bypass this prompt next time, supply `template`,

? Starter template: `tabs`

✓ Preparing directory `.\tabs` - done!

(node:22984) ExperimentalWarning: The `http2` module is an experimental API.

✓ Downloading and extracting `tabs` starter - done!

- Génère les différents fichiers requis par l'application et installe toutes les dépendances requises par le projet.

# Pour ajouter la plateforme Android au projet

```
C:\Users\Ousall\Desktop\AppIonic\tabs>ionic cordova platform add android --save
> ionic integrations enable cordova
[INFO] Downloading integration cordova
(node:21872) ExperimentalWarning: The http2 module is an experimental API.
[INFO] Copying integrations files to project
CREATE resources
CREATE resources\splash.png
CREATE resources\ios
CREATE resources\ios\splash
CREATE resources\ios\splash\Default~iphone.png
CREATE resources\ios\splash\Default@2x~universal~anyany.png
CREATE resources\ios\splash\Default@2x~iphone.png
```

# Si erreur Can not find module “@angular-devkit/build-angular”

- Voir les solutions proposées ici

<https://stackoverflow.com/questions/50333003/could-not-find-module-angular-devkit-build-angular>

If the following command does not work,

```
3 npm install --save-dev @angular-devkit/build-angular
```

then move to the project folder and run this command:

```
npm install --save @angular-devkit/build-angular
```

share improve this answer edited Jun 27 '18 at 15:07 grg 2,450 ● 2 ● 19 ● 33 answered May 20 '18 at 6:58 Murugaraju Perumalla 31 ● 3

# Si erreur Cannot find module '@angular/compiler'

- Voir les solutions proposées ici  
<https://stackoverflow.com/questions/42585663/cannot-find-module-angular-compiler>
  - npm uninstall angular-cli
  - npm install @angular/cli --save-dev

# Tabs App

- Si vous souhaitez utiliser le modèle d'onglets Ionic, l'application sera construite avec le menu d'onglets, l'en-tête et quelques écrans et fonctionnalités utiles. C'est le modèle Ionic par défaut.
- Ouvrez votre fenêtre de commande et choisissez où vous voulez créer votre application.

La commande Ionic **Start** va créer un répertoire nommé **test1** et créer et configurer les fichiers et dossiers Ionic.

```
C:\Users\OSall>cd Desktop
```

```
C:\Users\OSall\Desktop>md AppIonic
```

```
C:\Users\OSall\Desktop>cd AppIonic
```

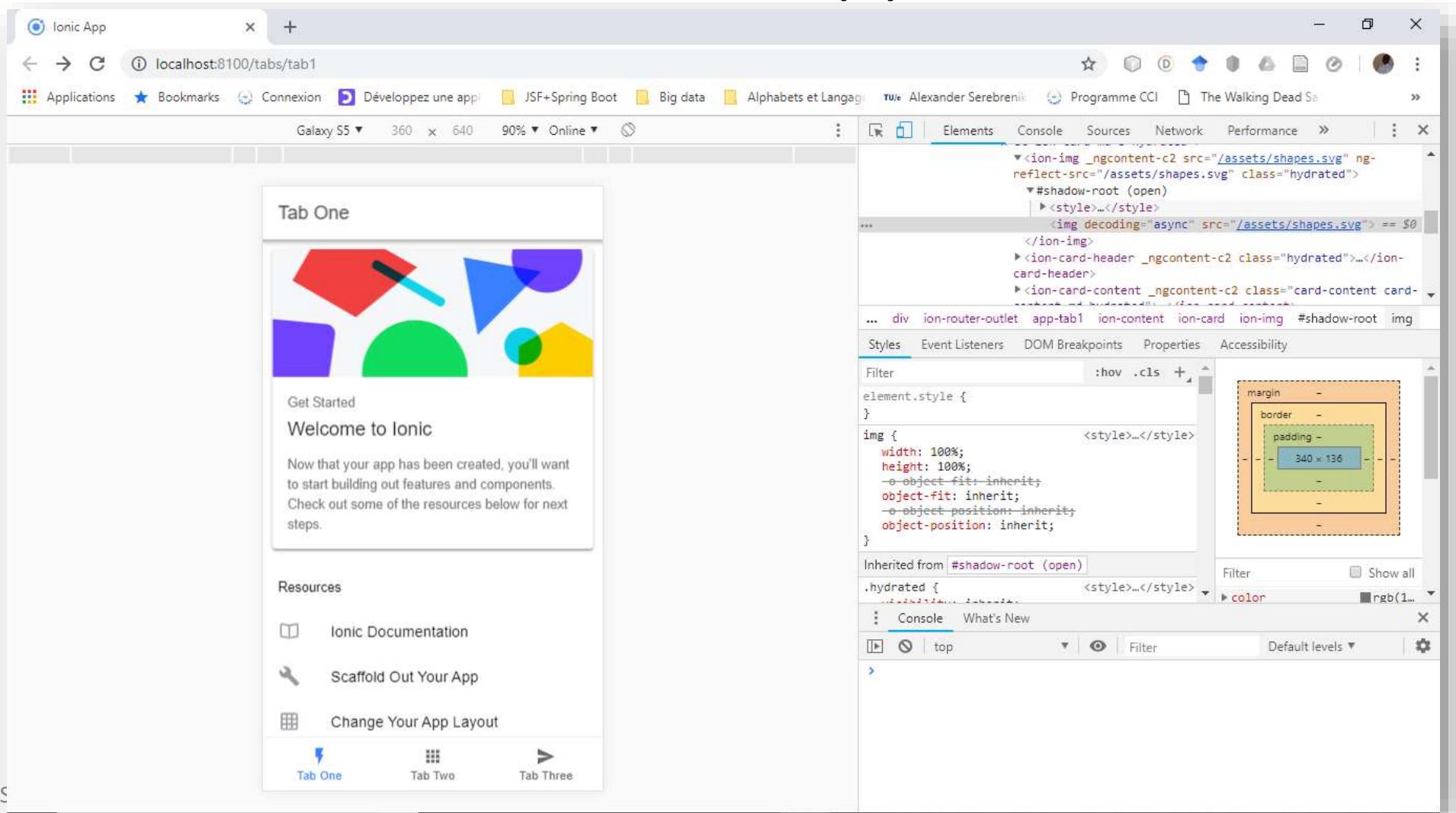
```
C:\Users\OSall\Desktop\AppIonic>ionic start test1 tabs
```

Pour lancer l'application:

```
C:\Users\OSall\Desktop\AppIonic>cd test1
```

```
C:\Users\OSall\Desktop\AppIonic\test1>ionic serve
```

# Tabs App: dans un navigateur :F12 pour accéder à la console dévelopeur



# Tabs App: pour tester sous émulateur android

- Ajouter la plateforme android au projet avec la commande

```
C:\Users\OSall\Desktop\AppIonic\test1>ionic cordova platform add android
```

- Lancer la commande suivante

```
C:\Users\OSall\Desktop\AppIonic\test1>ionic build android
```

- La dernière étape du processus d'installation consiste à exécuter votre application, qui lancera le périphérique mobile, s'il est connecté, ou l'émulateur par défaut, si aucun périphérique n'est connecté.

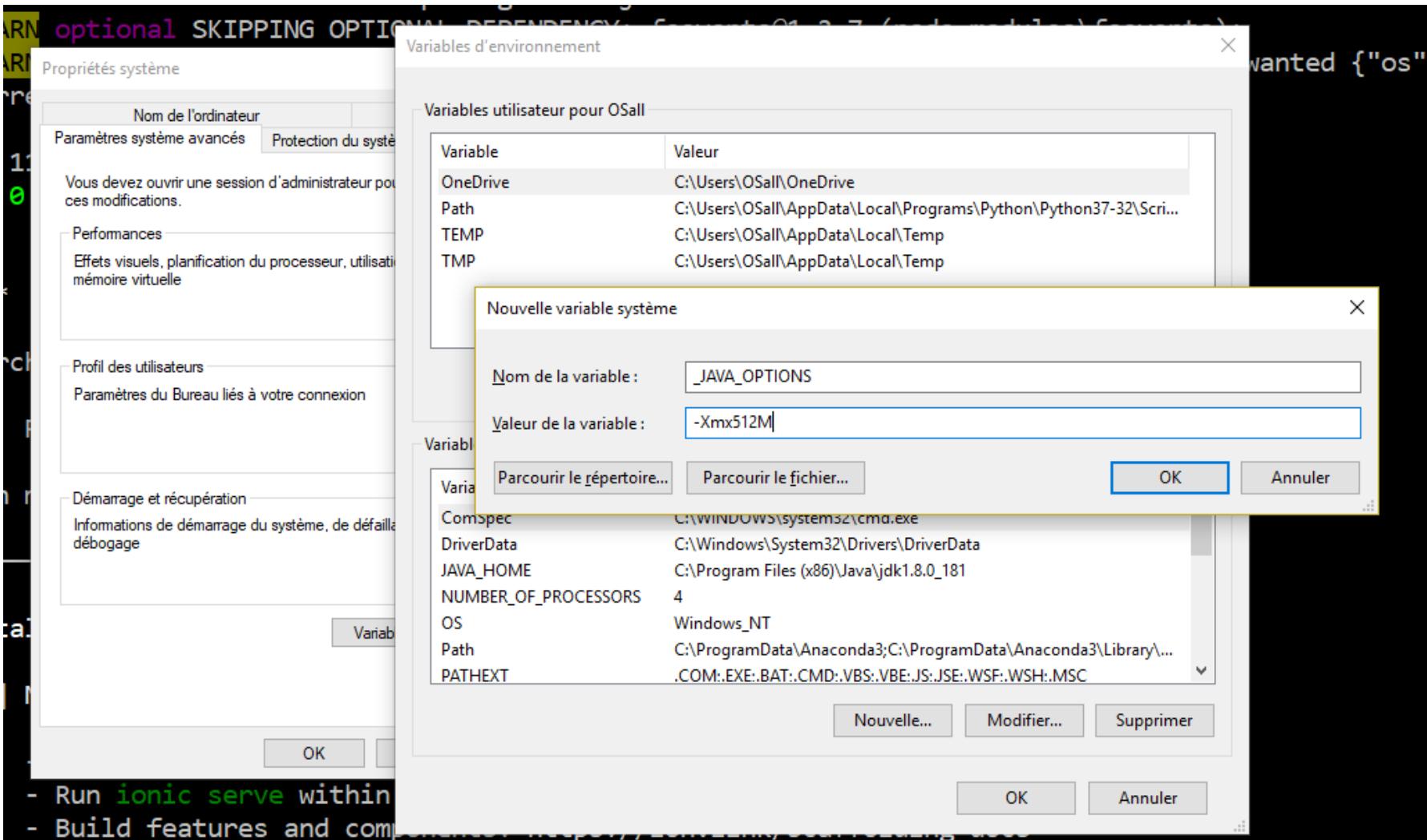
```
C:\Users\OSall\Desktop\AppIonic\test1>ionic cordova run android
```

- Option **-l** ou **-lab** pour que tout changement soit automatiquement appliqué dans le périphérique

```
c:\Users\OSall>ionic cordova run android -l
```

```
C:\Users\OSall\Desktop\AppIonic\test1>ionic serve --lab
```

Si erreur du genre Cordova Could not reserve enough space for XXKB object heap



# Si erreur du genre Cordova Could not reserve enough space for XXKB object heap

- <https://stackoverflow.com/questions/41216921/cordova-could-not-reserve-enough-space-for-2097152kb-object-heap>

```
args.push('-Dorg.gradle.jvmargs=-Xmx2048m')  
      to  
args.push('-Dorg.gradle.jvmargs=-Xmx1024m');
```

on the following location files.

1. project-folder\platforms\android\cordova\lib\builders\builders.js
2. project-folder\platforms\android\cordova\lib\builders\GradleBuilder.js
3. project-folder\platforms\android\cordova\lib\builders\StudioBuilder.js

[share](#) [improve this answer](#)

answered Jan 9 at 8:13



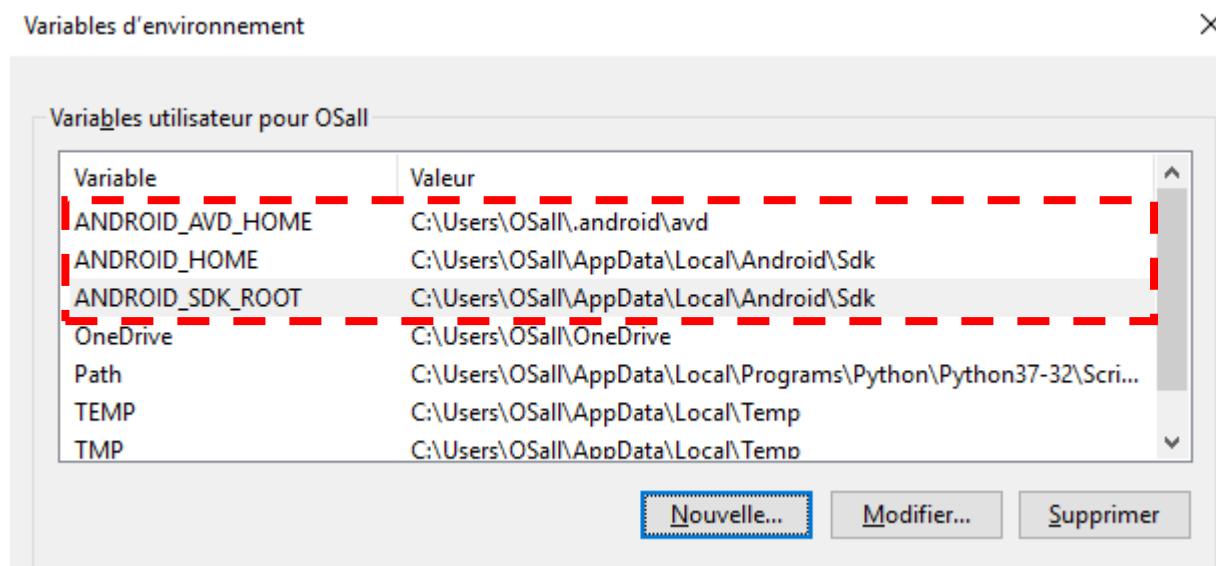
Abd Abughazaleh

44 • 11

Si erreur du genre Android Emulator Error  
Message: “PANIC: Missing emulator engine  
program for 'x86' CPUS.”

- <https://www.youtube.com/watch?v=feCIB6wdUyY>

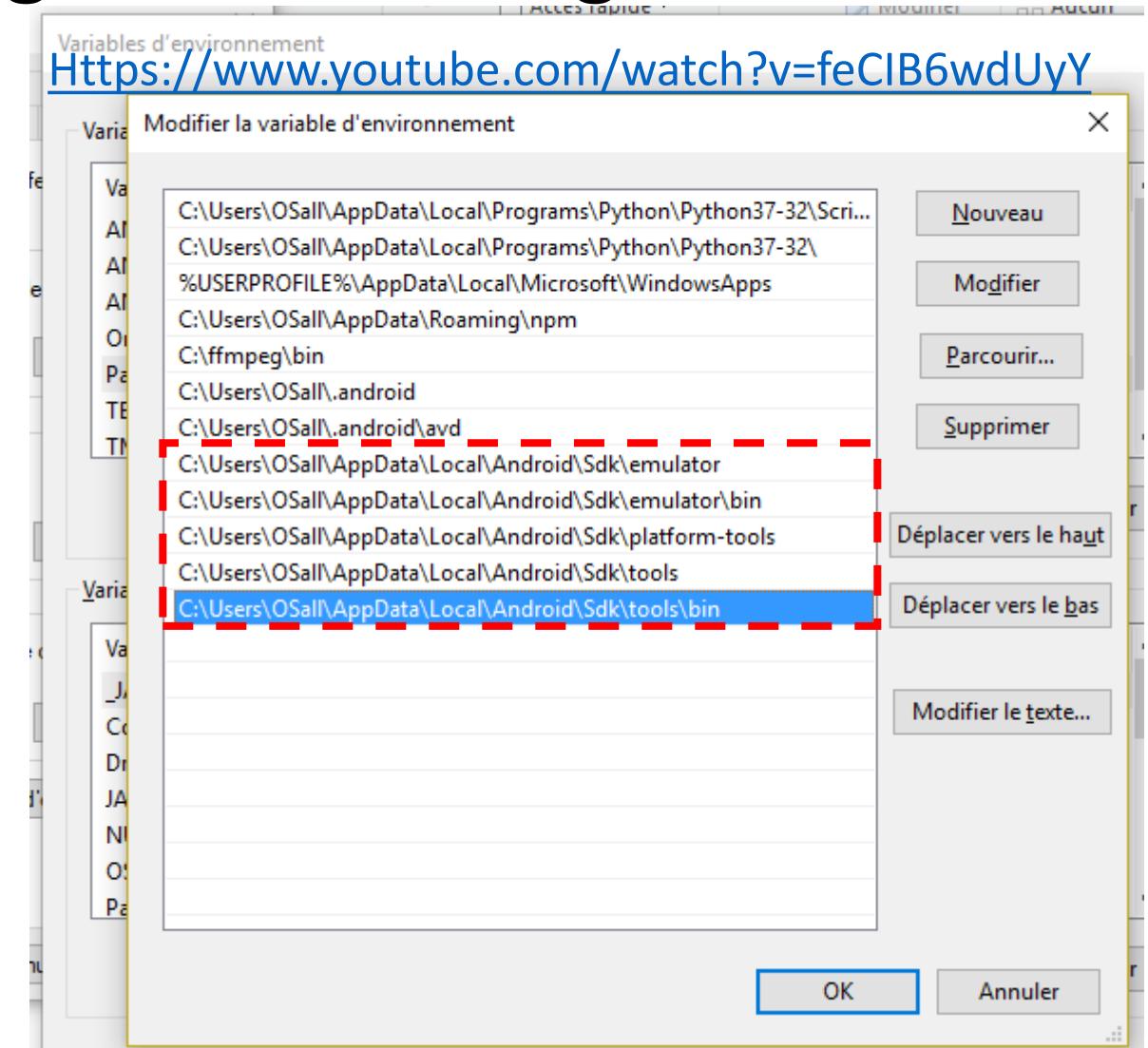
```
No target specified and no devices found, deploying to emulator
No emulator specified, defaulting to Android_Accelerated_x86_Oreo
Waiting for emulator to start...
PANIC: Missing emulator engine program for 'x86' CPU.
```



Vérifier que vous avez bien spécifié les variables d'environnement pour le SDK(ANDROID\_HOME et ANDROID\_SDK\_ROOT) et l'AVD(ANDROID\_AVD\_HOME)

# Si erreur du genre Android Emulator Error Message: “PANIC: Missing emulator engine program for 'x86' CPUS.”

Ensuite  
Vérifier que dans la variable  
**path** vous avez bien les  
chemins désignés par  
l'encadrée rouge ci-contre

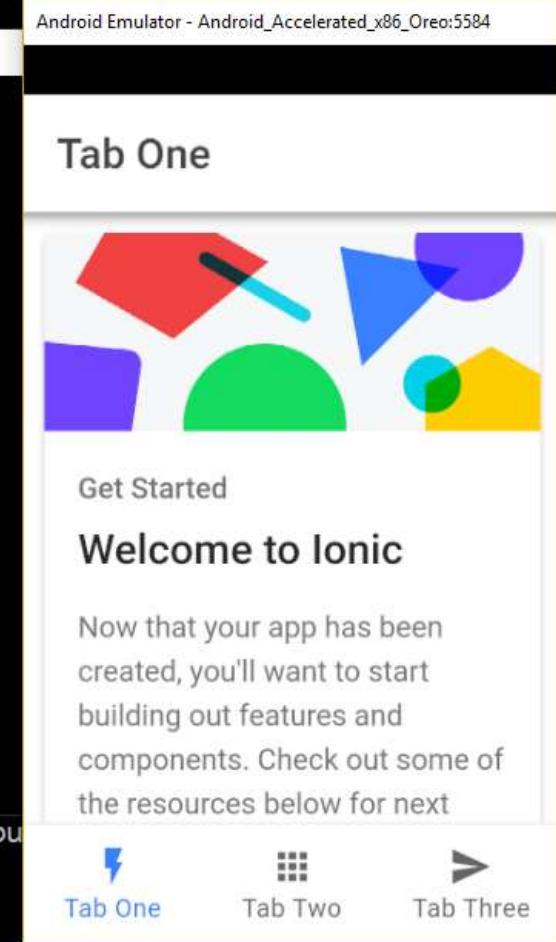


Si erreur du type: npm WARN deprecated circular-json@0.5.9: CircularJSON is in maintenance only, flattened is its successor. npm ERR! Unexpected end of JSON input while parsing near  
'...r/core":"4.4.0"}, "dir'

- Erreur arrivant en essayant de créer un nouveau projet
- Essayer d'exécuter cette ligne de commande :

**npm cache clean --force**

# Tabs App: pour tester sous émulateur android



The screenshot shows a terminal window on the left and an Android emulator window on the right.

**Terminal Output (Left):**

```
cmd: npm
:app:cdvBuildDebug UP-TO-DATE

BUILD SUCCESSFUL in 5s
46 actionable tasks: 1 executed, 45 up-to-date
Built the fd
C:\U
ANDROID_HOME
JAVA_HOME=C:
No target sp
No emulator
Waiting for
emulator: Re
HAX is worki
Your emulat
- Start And
- Select me
- Click "SD
- Check "Ar
- Click "OK

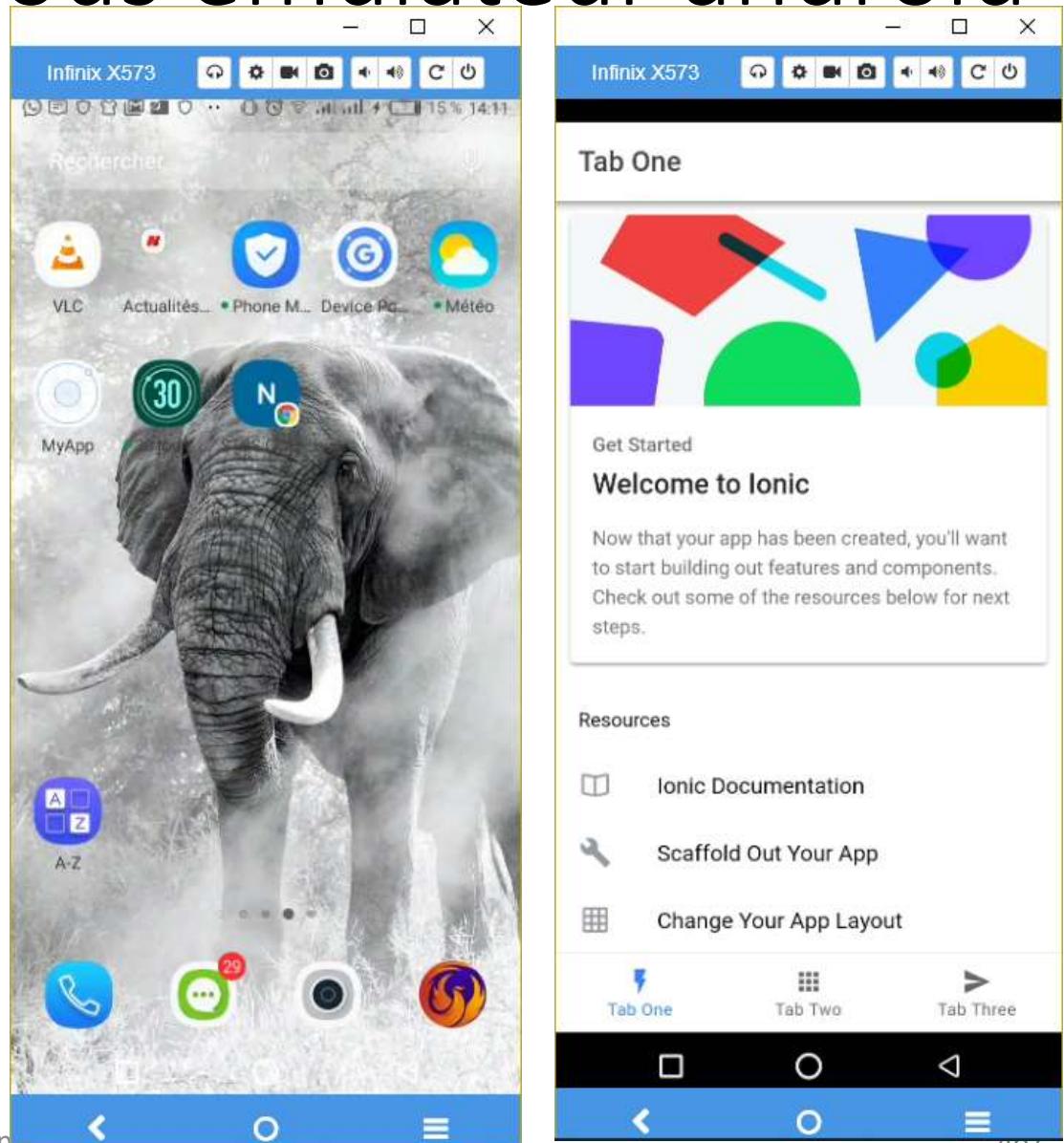
Waiting for
none
Skipping bui
Built the fd
C:\U
Using apk: C:\Users\OSall\Desktop\AppIonic\test1\platforms\android\app\build\output\apk\app-debug.apk
Package name: io.ionic.starter
INSTALL SUCCESS
LAUNCH SUCCESS
Proj
```

**Emulator Window (Right):**

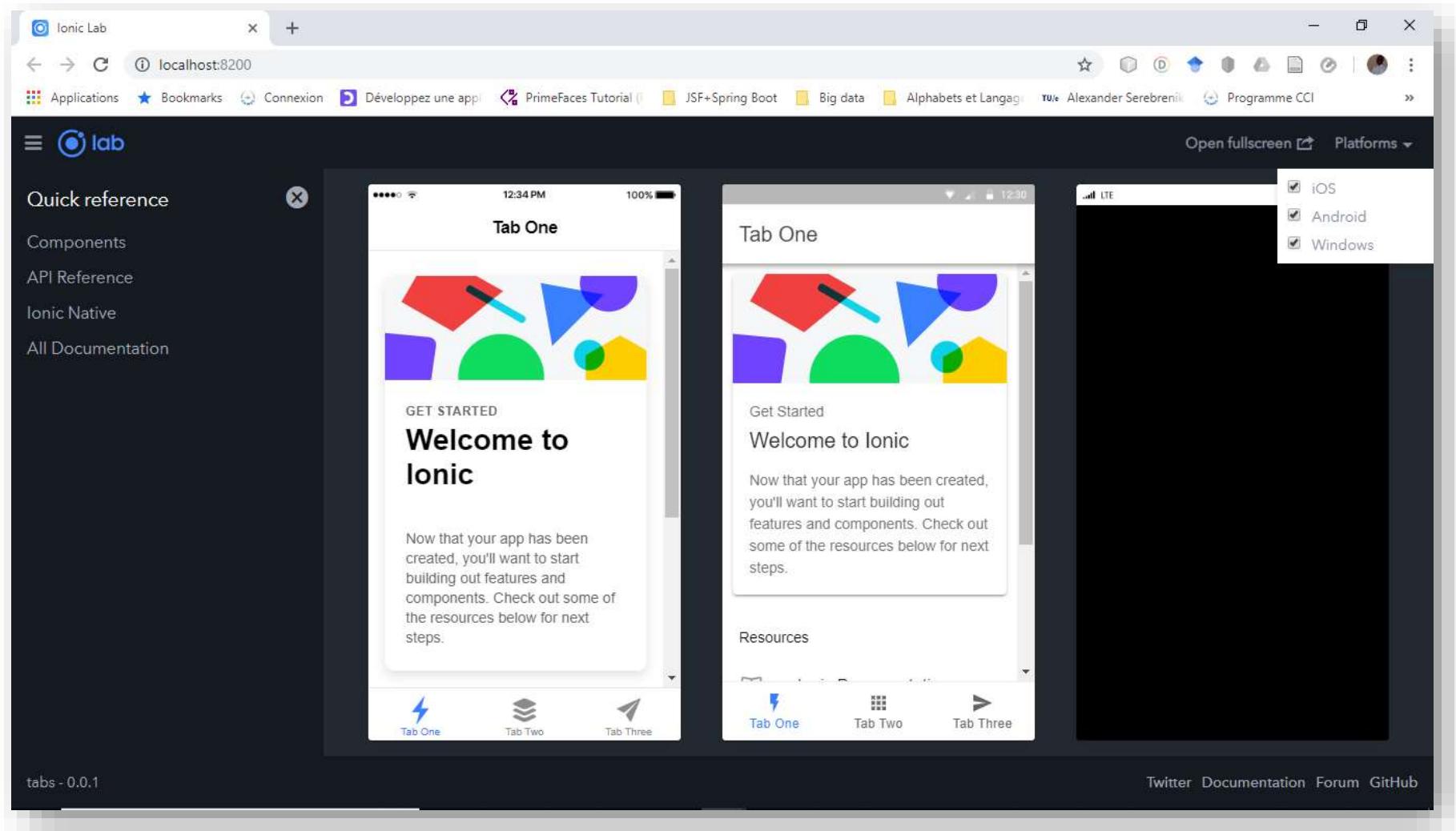
The emulator displays a tabs application titled "Tab One". The screen shows a "Welcome to Ionic" message with a "Get Started" button and some colorful geometric shapes. The bottom navigation bar features three tabs: "Tab One" (selected), "Tab Two", and "Tab Three". On the right side of the emulator window, there is a vertical toolbar with various icons for device control.

# Tabs App: pour tester sous émulateur android

- Installer Vysor sur votre téléphone et la machine pour lancer sur votre téléphone portable, c'est plus rapide que l'émulateur.
- Vysor va vous permettre de visualiser sur l'ordinateur l'écran de votre téléphone.



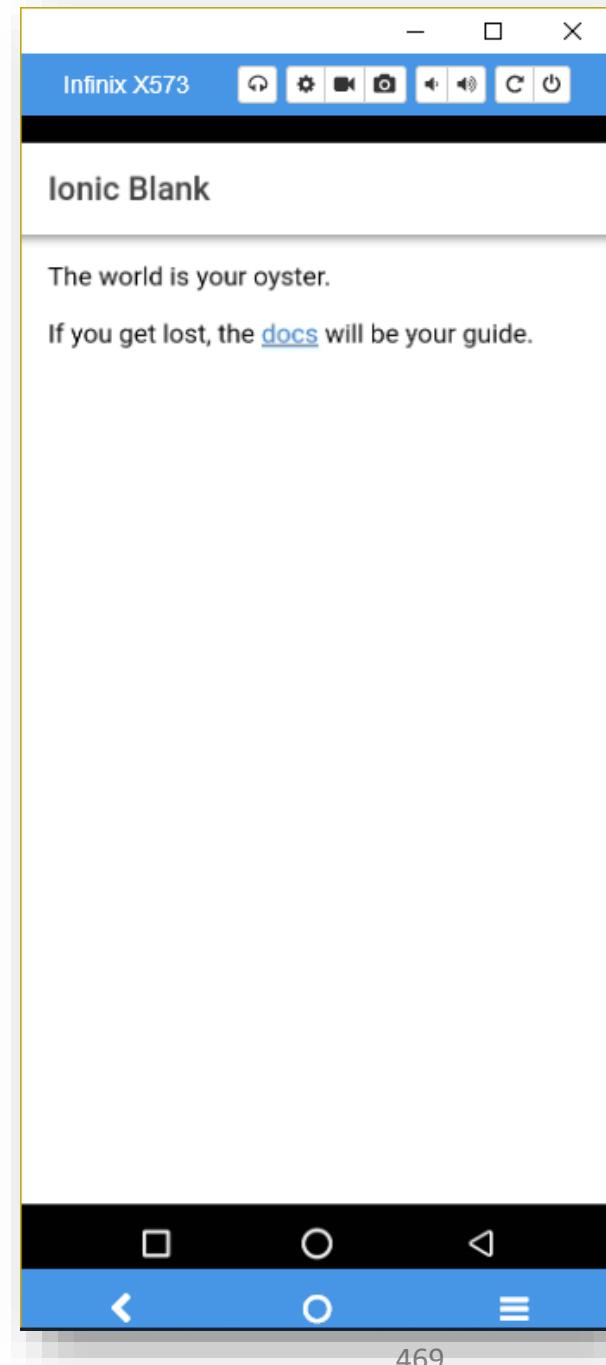
# Exécution en mode lab avec *ionic serve --l*



# Blank App

- Si vous souhaitez recommencer à zéro, vous pouvez installer le modèle vierge(Blank) Ionic. Utiliser la commande **ionic start myApp blank**

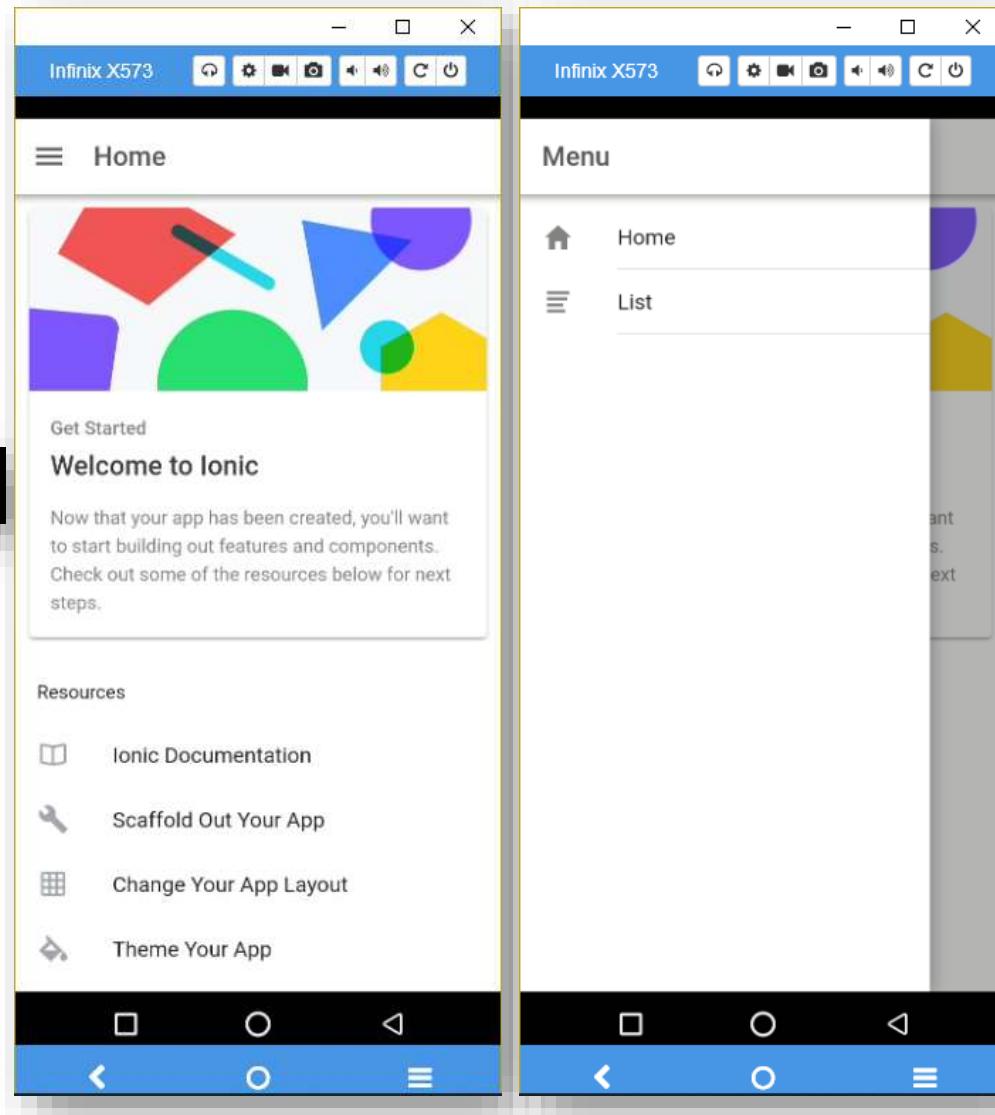
```
C:\Users\OSall\Desktop\AppIonic>ionic start test2 blank
```



# Sidemenu App

- Pour créer un projet avec le template **sidemenu**. Utiliser la commande ci-dessous

```
C:\Users\OSall\Desktop\AppIonic>ionic start test3 sidemenu
```

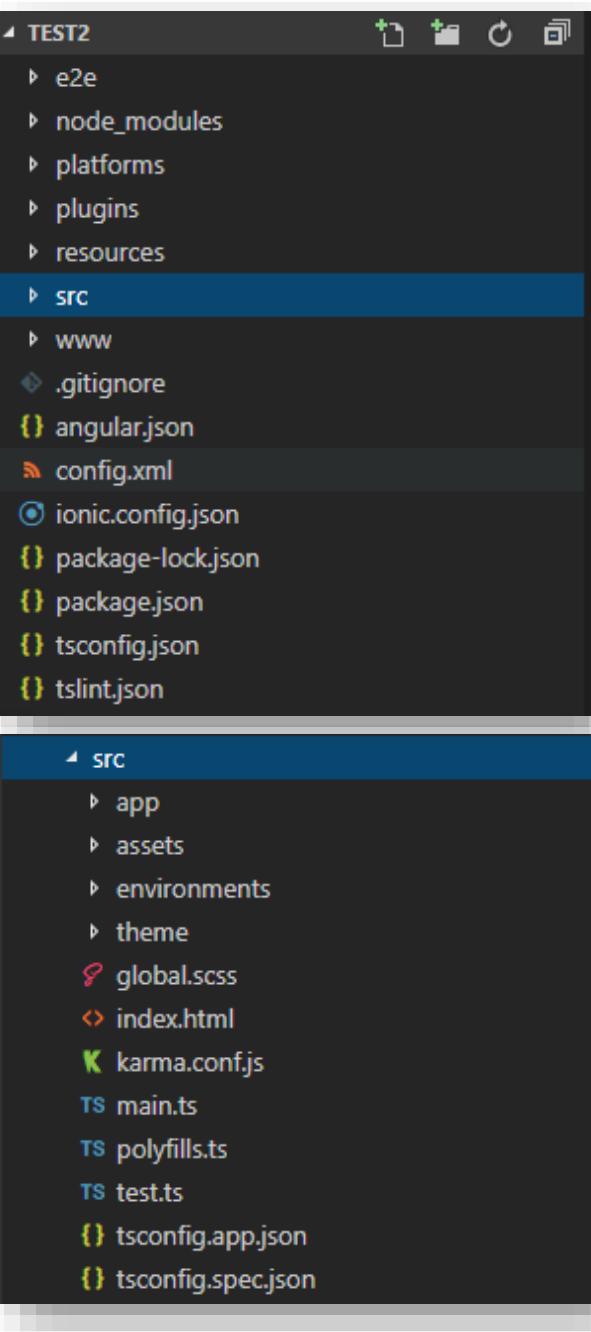


# Structure d'un projet Ionic 4

Ouvrir le projet test2 par exemple sous Visual Studio Code ou bien exécuter

Dans le projet la commande « **code .** »

- Le dossier **node\_modules** est automatiquement généré une fois que vous avez installé les dépendances **npm** avec «**npm install**» (Ionic l'avait déjà fait pour vous au début). Cette commande analysera le fichier **package.json** pour tous les packages devant être installés. Il s'agit d'un fichier Node.js classique. Dossier contenant les dépendances externes du projets (Librairies Java Script et CSS)
- Le dossier **platforms** contient vos projets natifs. Vous devez d'abord les ajouter (nous verrons plus tard) et ils seront générés dans ce dossier.

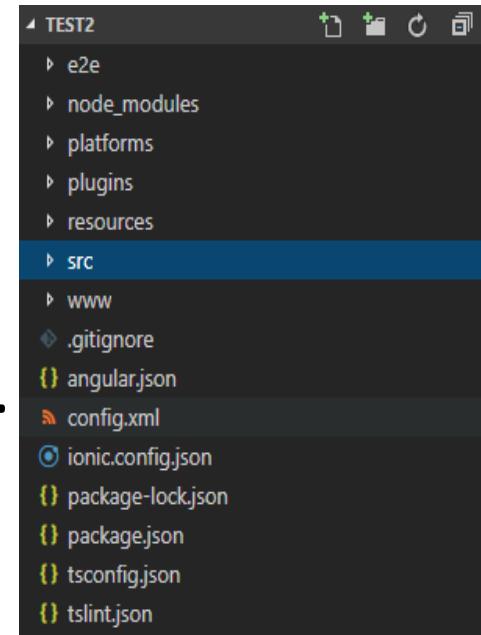


# Structure d'un projet Ionic

Ouvrir le projet test2 par exemple sous Visual Studio Code ou bien exécuter

Dans le projet la commande « **code .** »

- Le dossier **plugins** contient tous les plugins que vous avez installés.
- Le dossier **resources** est un dossier d'Ionic pouvant contenir l'icône de votre application et votre écran de démarrage.
- Le dossier **www** est le dossier de travail généré par angular.
- Les deux fichiers **tsconfig.json** et **tslint.json** sont relatifs à TypeScript et comment il est compilé. Vous pouvez également y faire quelques manipulations pour rendre moins strict les règles de gestion des erreurs.

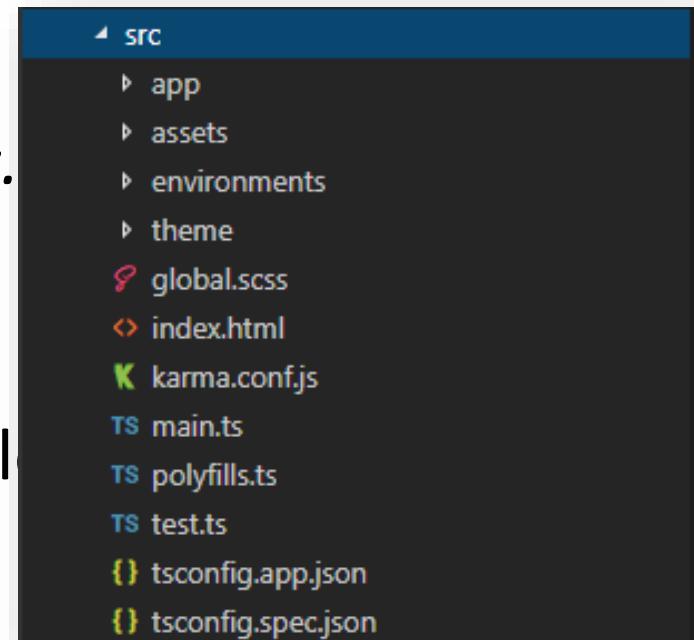


# Structure d'un projet Ionic

Ouvrir le projet test2 par exemple sous Visual Studio Code ou bien exécuter

Dans le projet la commande « **code .** »

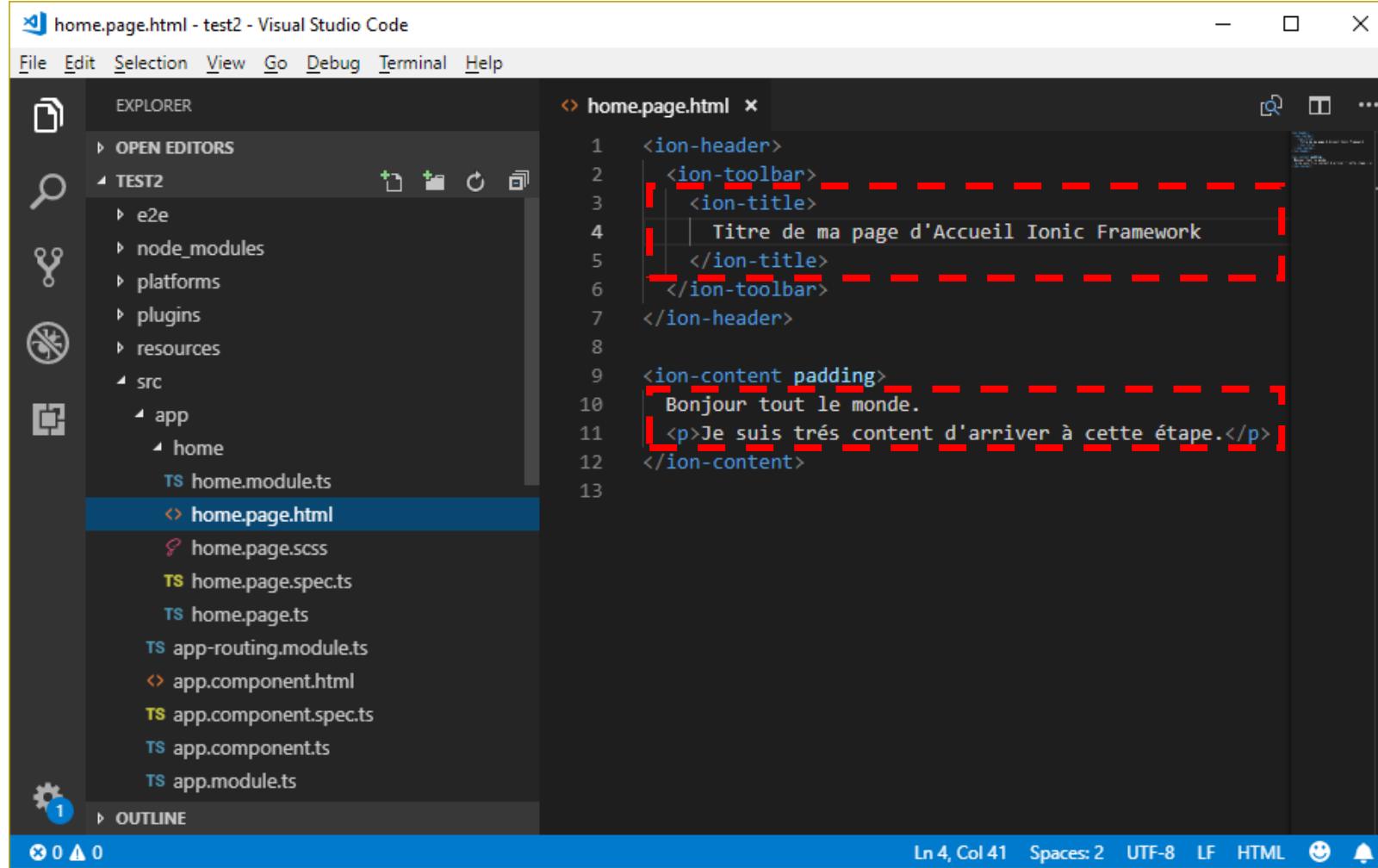
- Le dossier **src** est le dossier le plus important, et 99% de votre travail se déroulera dans ce dossier. C'est le dossier qui contiendra le code source de votre application : *Composants web, services, modèle, etc.*
- **app** : contient la logique applicative du projet: composants, services...
- **assets** : Ressources statiques de l'application mobile (images, icônes, etc.)



# Structure d'un projet Ionic: **Les fichiers de configuration**

- **config.xml** : Fichier de configuration de l'application. Dans ces premières lignes de codes, nous avons:
  - le nom de l'application `<name></name>`
  - la description de l'application `<description></description>` et l'auteur de l'application `<author></author>`.
  - le point d'entrée dans l'application (souvent index.html). `<content src="index.html"/>`
- **package.json** : fichier déclaratif des dépendances du projet

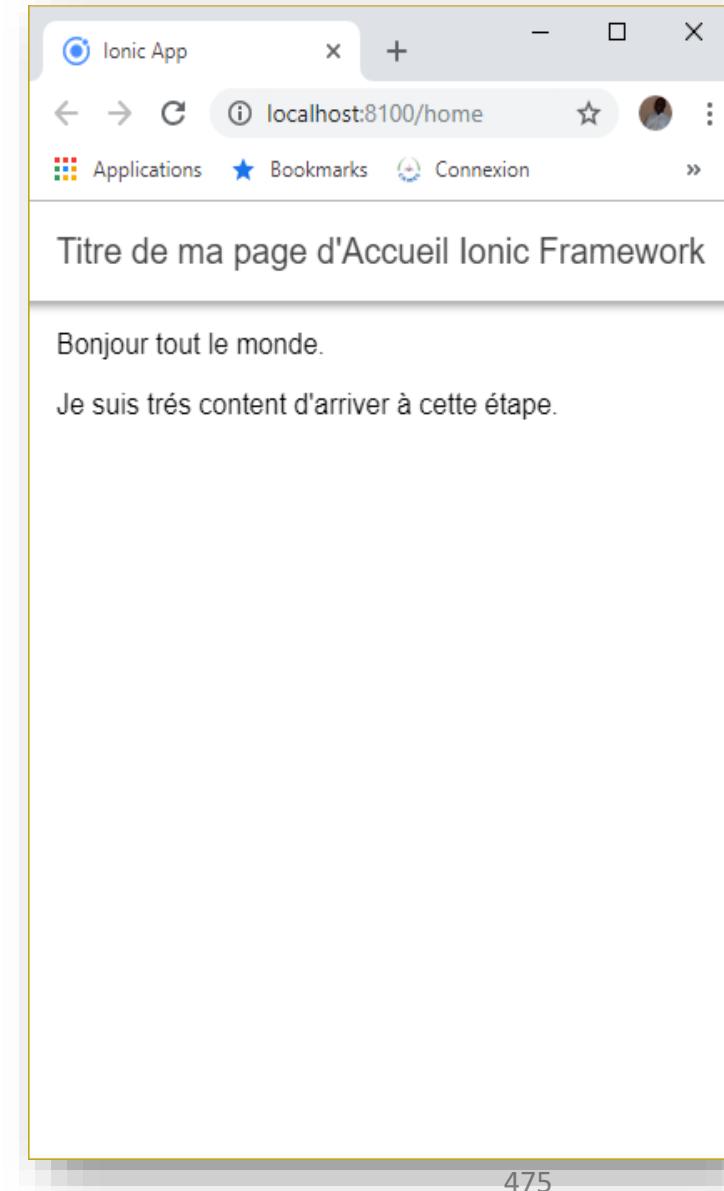
# Ajouter du contenu à notre application: lancer le projet de type blank test2



The screenshot shows the Visual Studio Code interface with the following details:

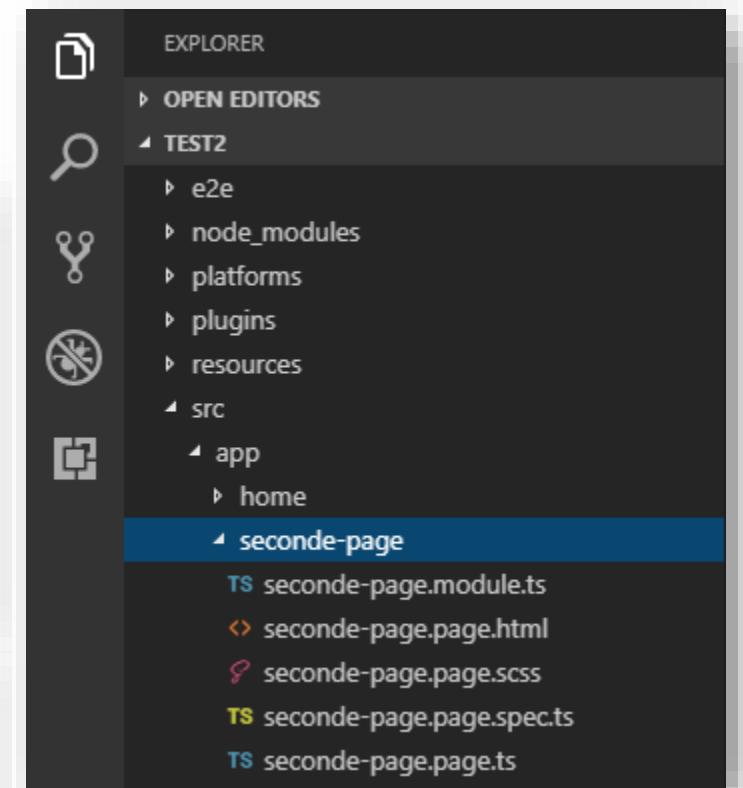
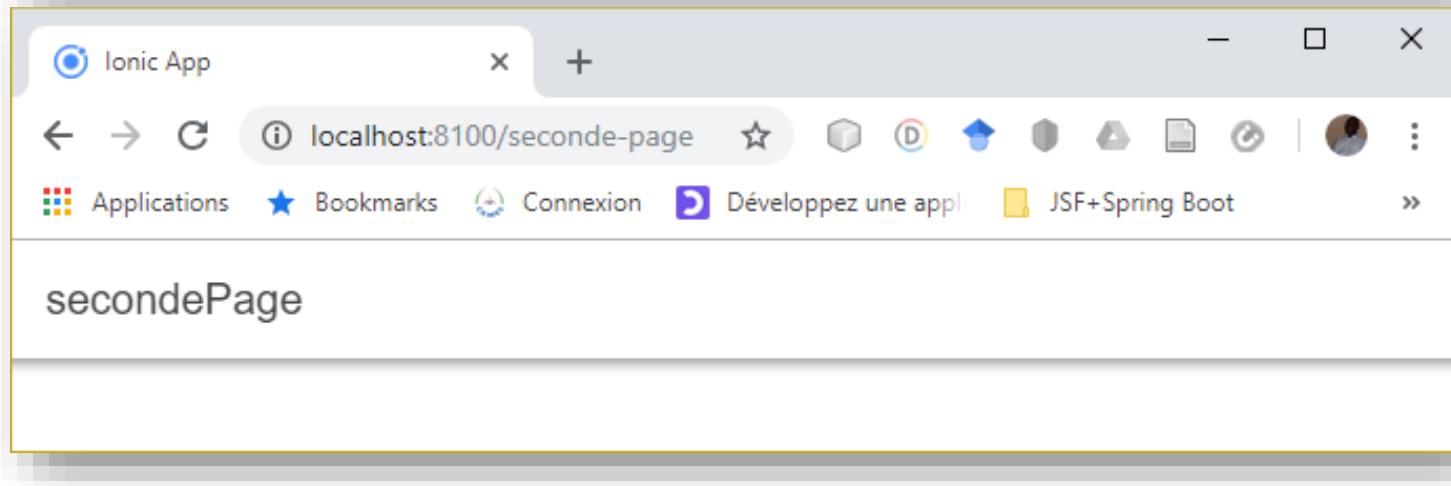
- Title Bar:** home.page.html - test2 - Visual Studio Code
- File Menu:** File Edit Selection View Go Debug Terminal Help
- Explorer:** TEST2 folder structure:
  - e2e
  - node\_modules
  - platforms
  - plugins
  - resources
  - src
    - app
      - home
        - home.module.ts
        - home.page.html
        - home.page.scss
        - home.page.spec.ts
        - home.page.ts
      - app-routing.module.ts
      - app.component.html
      - app.component.spec.ts
      - app.component.ts
      - app.module.ts
- Editor:** home.page.html content:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Titre de ma page d'Accueil Ionic Framework
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10   Bonjour tout le monde.
11   <p>Je suis très content d'arriver à cette étape.</p>
12 </ion-content>
```
- Status Bar:** Ln 4, Col 41 Spaces: 2 UTF-8 LF HTML

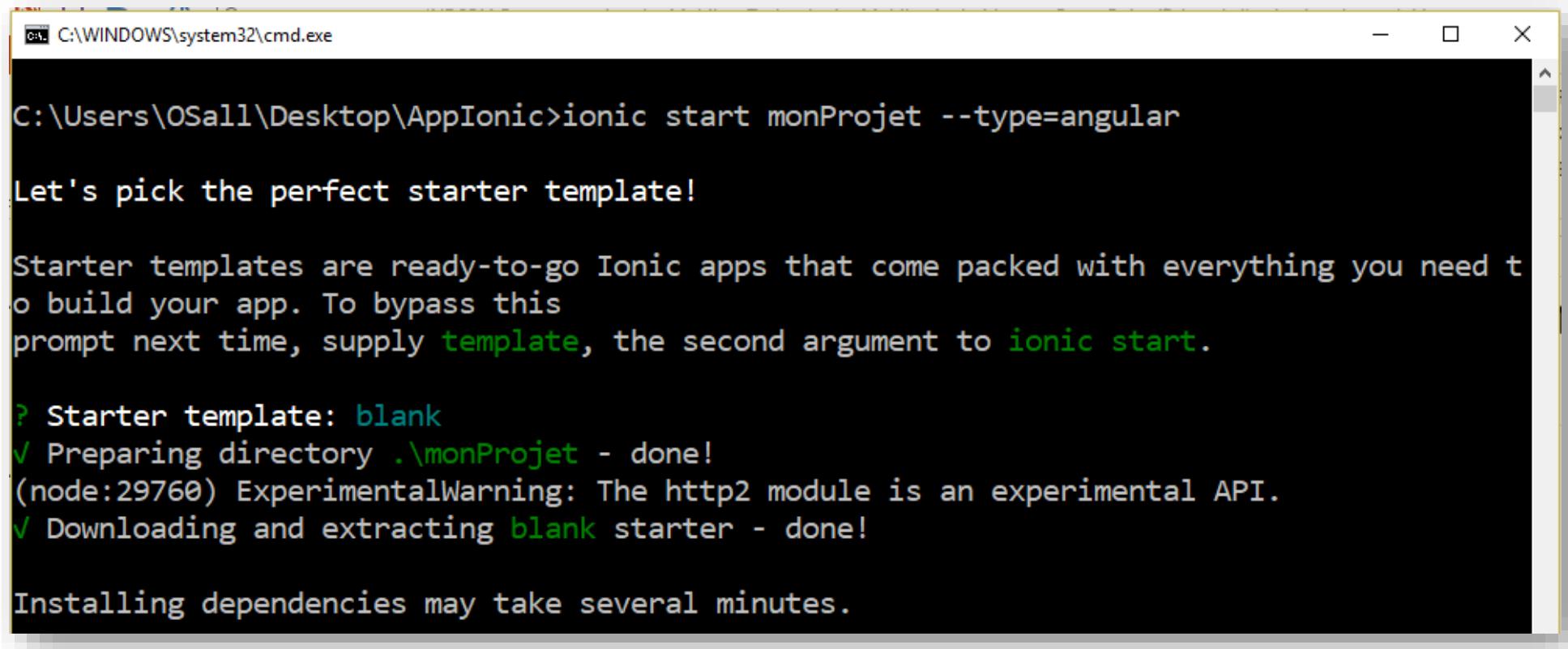


# Ajouter du contenu à notre application: ajouter une nouvelle page

```
C:\Users\OSall\Desktop\AppIonic\test2>ionic g page secondePage
> ng generate page secondePage
CREATE src/app/seconde-page/seconde-page.module.ts (569 bytes)
CREATE src/app/seconde-page/seconde-page.page.html (138 bytes)
CREATE src/app/seconde-page/seconde-page.page.spec.ts (727 bytes)
CREATE src/app/seconde-page/seconde-page.page.ts (279 bytes)
CREATE src/app/seconde-page/seconde-page.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (474 bytes)
[OK] Generated page!
```



# Ajouter du contenu: Créer un projet nommé monProjet de type blank



```
C:\Users\OSall\Desktop\AppIonic>ionic start monProjet --type=angular
Let's pick the perfect starter template!
Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your app. To bypass this prompt next time, supply template, the second argument to ionic start.
? Starter template: blank
✓ Preparing directory .\monProjet - done!
(node:29760) ExperimentalWarning: The http2 module is an experimental API.
✓ Downloading and extracting blank starter - done!

Installing dependencies may take several minutes.
```

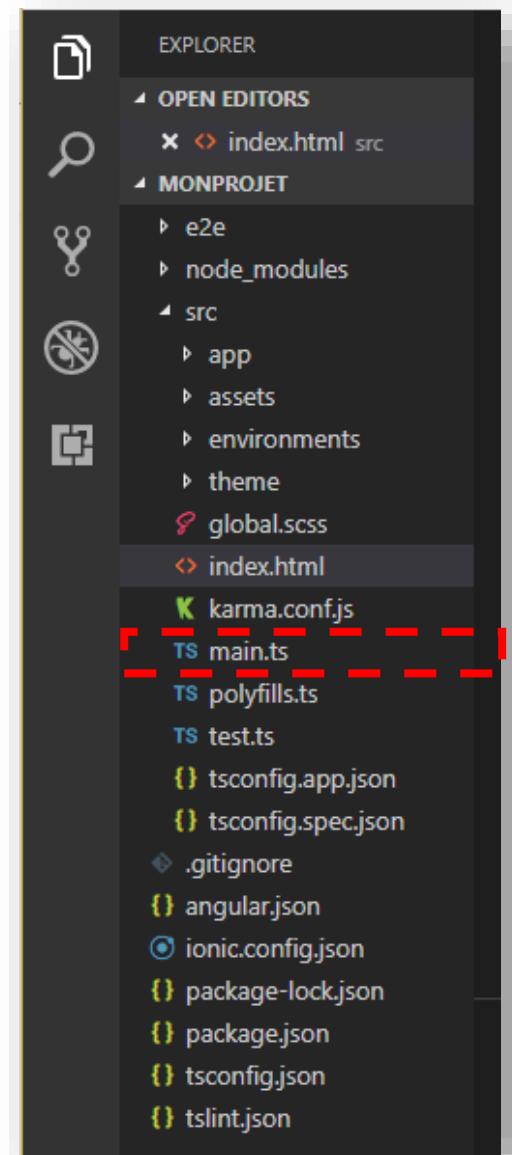
- Vous pouvez le lancer en mode lab avec ***ionic serve -Lab***

# Structure du projet: main.ts

- Point d'entrée de l'application

```
TS main.ts  x
1 import { enableProdMode } from '@angular/core';
2 import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
3
4 import { AppModule } from './app/app.module';
5 import { environment } from './environments/environment';
6
7 if (environment.production) {
8   enableProdMode();
9 }
10
11 platformBrowserDynamic().bootstrapModule(AppModule)
12   .catch(err => console.log(err));
```

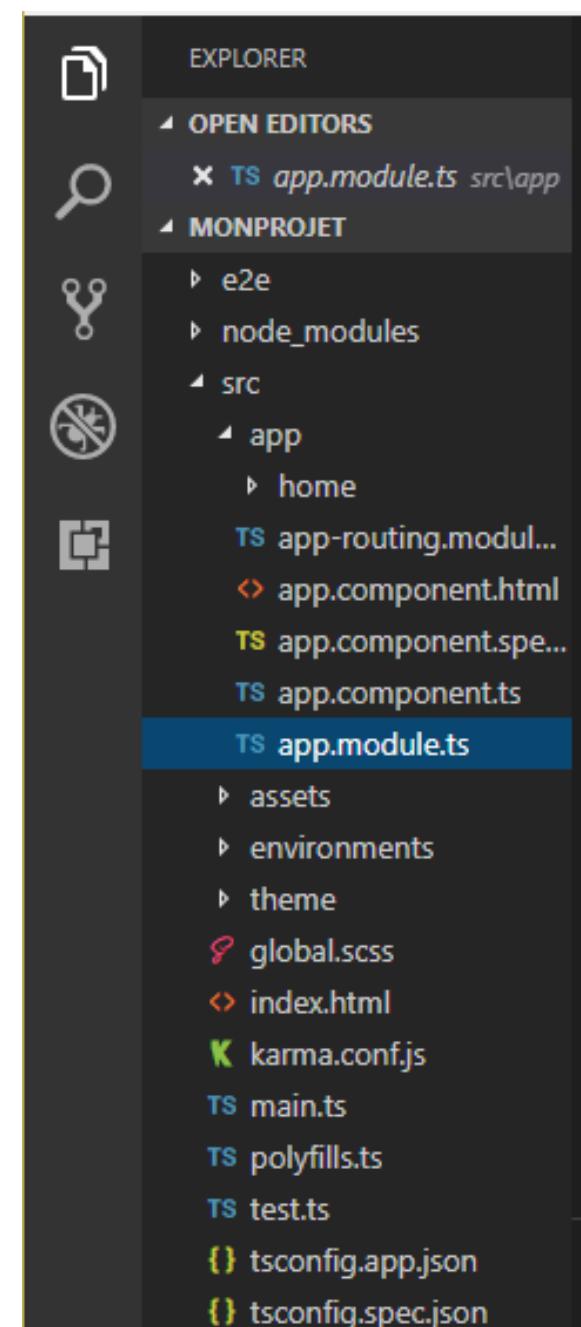
**AppModule** sera celui qui va démarrer en premier parmi les modules. Il s'agit de la classe **AppModule** dans le fichier **app.module.ts** du dossier **App**



# Structure du projet: app.module.ts

```
TS app.module.ts ×  
1 import { NgModule } from '@angular/core';  
2 import { BrowserModule } from '@angular/platform-browser';  
3 import { RouteReuseStrategy } from '@angular/router';  
4  
5 import { IonicModule, IonicRouteStrategy } from '@ionic/angular';  
6 import { SplashScreen } from '@ionic-native/splash-screen/ngx';  
7 import { StatusBar } from '@ionic-native/status-bar/ngx';  
8  
9 import { AppComponent } from './app.component';  
10 import { AppRoutingModule } from './app-routing.module';  
11  
12 @NgModule({  
13   declarations: [AppComponent],  
14   entryComponents: [],  
15   imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule],  
16   providers: [  
17     StatusBar,  
18     SplashScreen,  
19     { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }  
20   ],  
21   bootstrap: [AppComponent]  
22 })  
23 export class AppModule {}
```

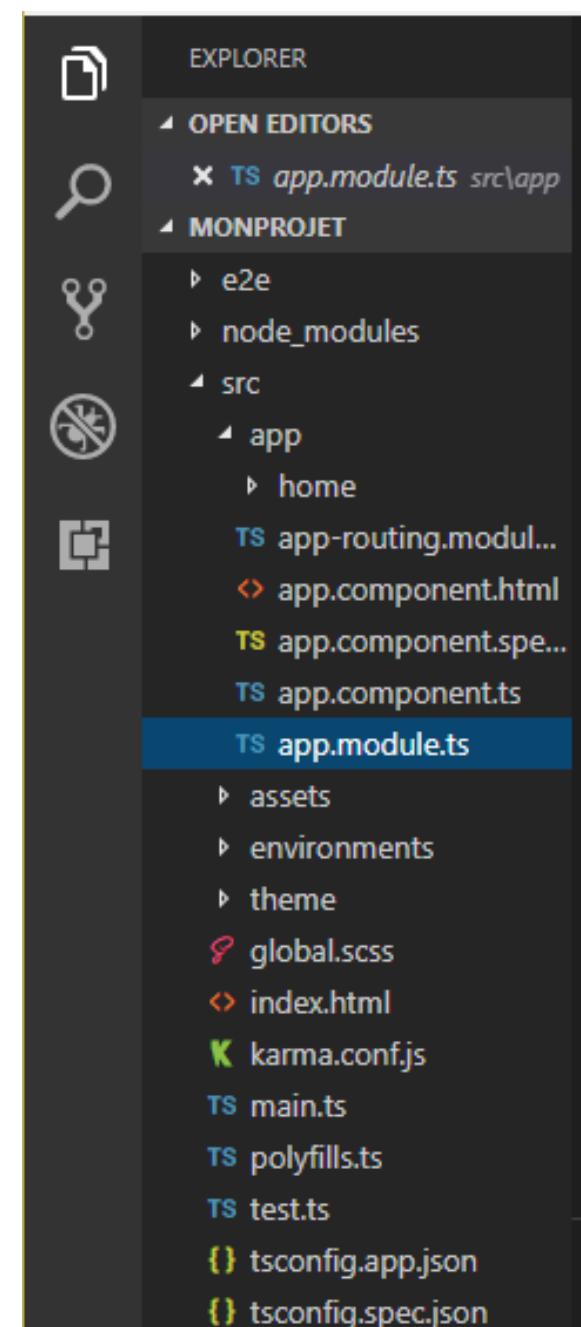
**bootstrap**: composant racine créé et inséré par Angular dans la page Web hôte index.html.



# Structure du projet: `app.module.ts`

- Le décorateur **@NgModule** identifie AppModule en tant que classe NgModule. @NgModule prend un objet de métadonnées qui indique à Angular comment compiler et lancer l'application.
  - **declarations**: la classe représentant le module. Angular a trois types de classes de modules : **components**, **directives**, and **pipes**..
  - **imports**: importez des modules comme BrowserModule pour disposer de services spécifiques au navigateur, tels que le rendu DOM, la désinfection et la localisation.
  - **providers** - Pour déclarer les fabriques de services.
- **bootstrap**: composant racine créé et inséré par Angular dans la page Web hôte index.html. Le composant qui va démarrer en premier

L'application par défaut créée ne comporte qu'un seul composant, **AppComponent**. Elle figure donc à la fois dans les tableaux de déclarations et dans les tableaux d'amorçage.



index.html - monProjet - Visual Studio Code

File Edit Selection View Go Debug Terminal Help

EXPLORER

OPEN EDITORS

- index.html src

MONPROJET

- e2e
- node\_modules
- src
  - app
  - assets
  - environments
  - theme
  - global.scss
- index.html
- karma.conf.js
- main.ts
- polyfills.ts
- test.ts
- { tsconfig.app.json
- { tsconfig.spec.json
- .gitignore
- { angular.json
- ionic.config.json
- { package-lock.json
- { package.json
- { tsconfig.json
- { tslint.json

PROBLEMS OUTPUT DEBUG CONSOLE

External: http://192.168.1.10:8100  
DevApp: monProjet@8100 on I

Use Ctrl+C to quit this process

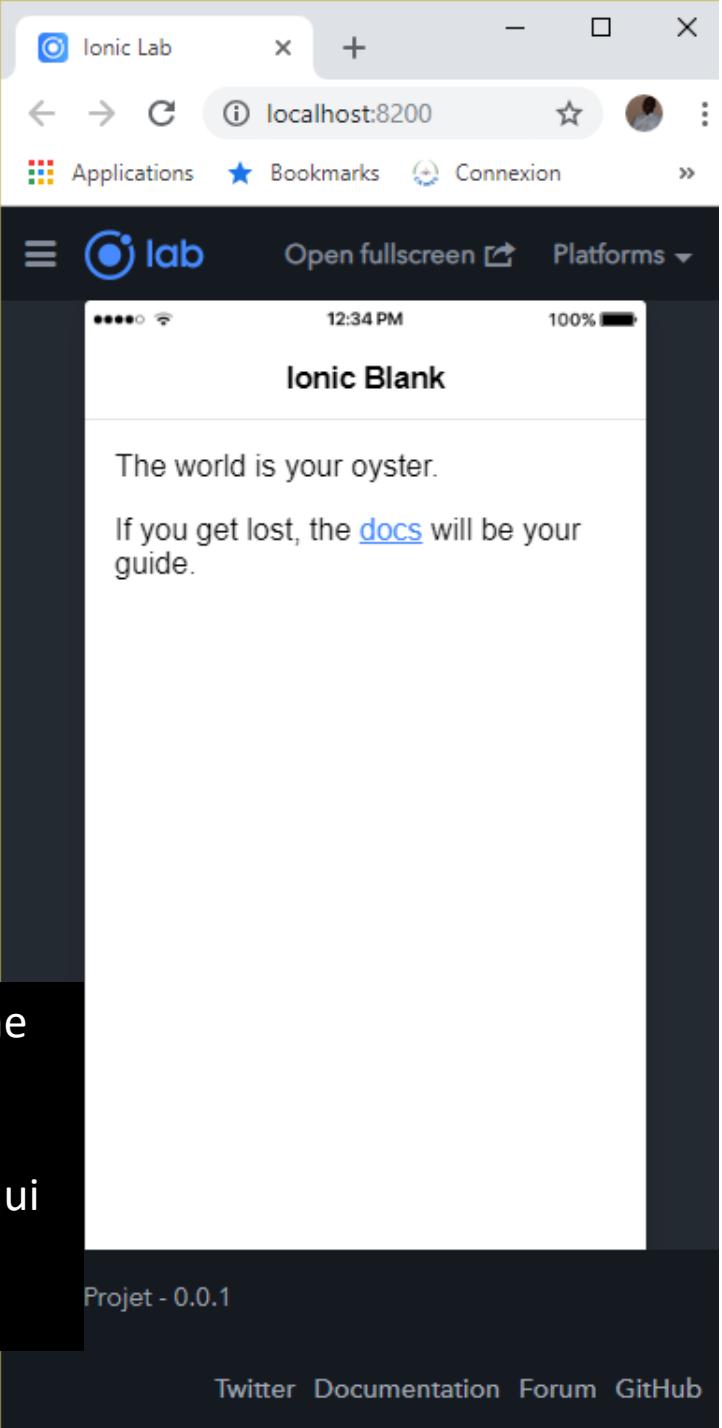
[INFO] Browser window opened to http://192.168.1.10:8100/index.html

Ln 1, Col 1 Spaces: 2 UTF-8 LF HTML

0 0 0

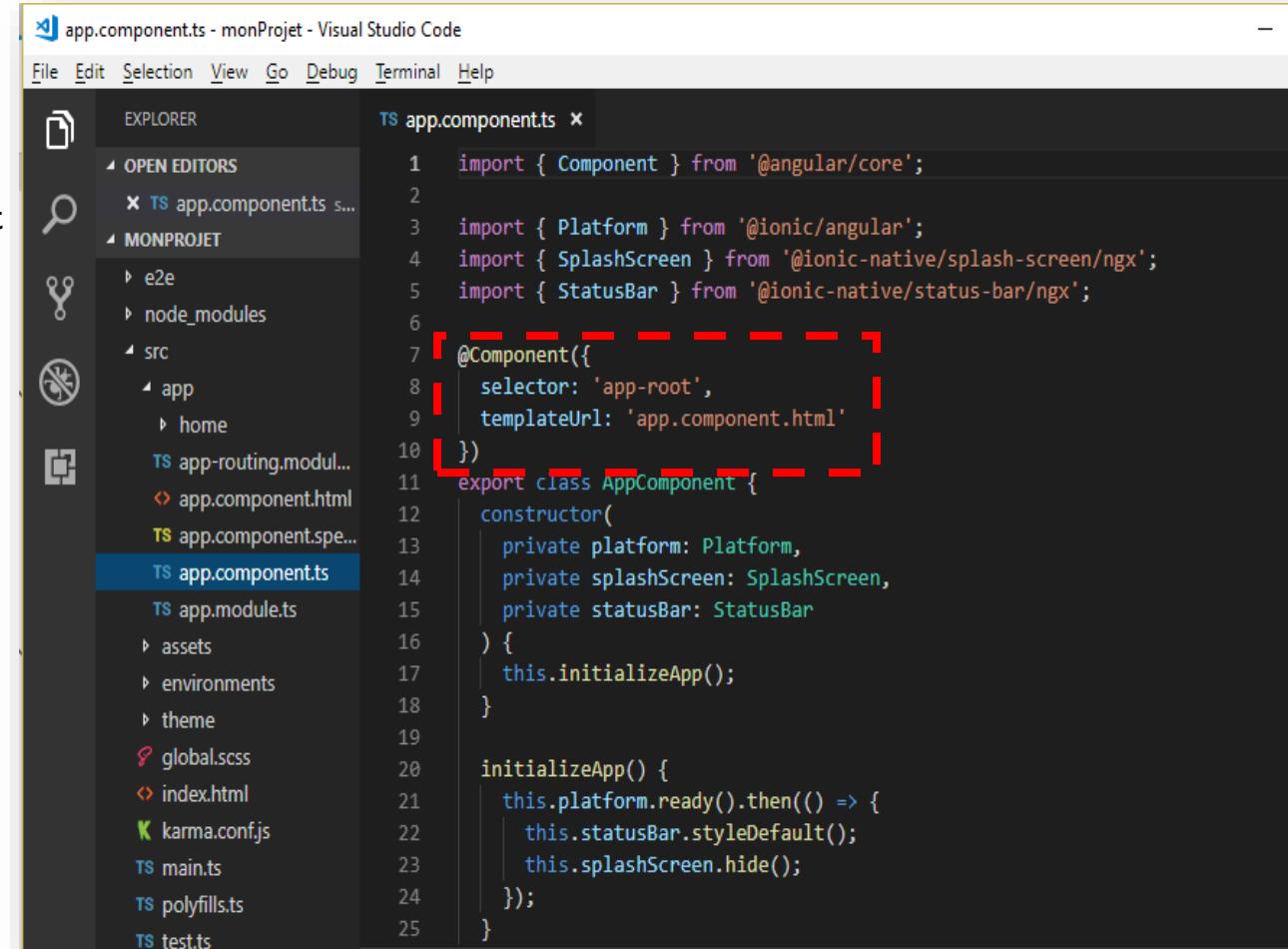
The code editor shows the content of index.html. The file starts with a standard HTML header and meta tags. It includes viewport settings and links to assets like a favicon. The body contains a single element: <app-root></app-root>. A red dashed rectangle highlights this entire body section.

La seule chose que nous affichons est une racine d'application, qui n'est pas encore très claire. Cette racine d'application est remplacée par le premier vrai code HTML de notre application, qui se trouve toujours dans le fichier app / app.component.html.



# Structure du projet: app.component.ts

- Un composant est une classe qui possède le décorateur `@Component`
- Ce décorateur possède les propriétés suivantes :
  - **selector** : indique la déclaration qui permet d'insérer le composant dans le document HTML. Cette déclaration peut être:
    - Le nom de la balise associé à ce composant
      - `selector:app-root`
      - Dans ce cas le composant sera inséré par : `<app-root></app-root>`
    - Le nom de l'attribut associé à ce composant:
      - `selector:[app-root]`
      - Dans ce cas le composant sera inséré par : `<div app-root></div>`
    - Le nom de la classe associé à ce composant:
      - `selector:.app-root`
      - Dans ce cas le composant sera inséré par : `<div class="app-root"></div>`
  - **template ou templateUrl** :
    - **template** : permet de définir dans à l'intérieur du décorateur le code HTML représentant la vue du composant
    - **templateUrl** : permet d'associer un fichier externe HTML contenant la structure de la vue du composant
  - **styleUrls** : spécifier les feuilles de styles CSS associées à ce composant



```
File Edit Selection View Go Debug Terminal Help
EXPLORER
OPEN EDITORS
MONPROJET
e2e
node_modules
src
app
home
app-routing.modul...
app.component.html
app.component.spe...
TS app.component.ts
TS app.module.ts
assets
environments
theme
global.scss
index.html
karma.conf.js
TS main.ts
TS polyfills.ts
TS test.ts
TS app.component.ts x
1 import { Component } from '@angular/core';
2
3 import { Platform } from '@ionic/angular';
4 import { SplashScreen } from '@ionic-native/splash-screen/ngx';
5 import { StatusBar } from '@ionic-native/status-bar/ngx';
6
7 @Component({
8   selector: 'app-root',
9   templateUrl: 'app.component.html'
10 })
11 export class AppComponent {
12   constructor(
13     private platform: Platform,
14     private splashScreen: SplashScreen,
15     private statusBar: StatusBar
16   ) {
17     this.initializeApp();
18   }
19
20   initializeApp() {
21     this.platform.ready().then(() => {
22       this.statusBar.styleDefault();
23       this.splashScreen.hide();
24     });
25 }
```

# Structure du projet: **app.component.html**

app.component.html

```
1  <ion-app>
2    <ion-router-outlet></ion-router-outlet>
3  </ion-app>
4
```

Ceci est la clé pour comprendre le fonctionnement du routage:  
Dire à Angular où vous souhaitez afficher les components dans  
le template lorsque l'utilisateur navigue vers la route en  
question selon les informations résolues pour un chemin.

# Structure du projet: **app-routing.module.ts**

La propriété path définit l'URL, et la propriété composant définit le composant devant être affiché par <ion-router-outlet> lorsque cette URL est atteinte. Si je devais aller à l'URL suivante: http://localhost:8100/home, la page d'accueil serait affichée. Nous avons également un itinéraire par défaut défini au bas de sorte que, si aucun itinéraire n'est fourni, il utilisera l'itinéraire de connexion.

TS app-routing.module.ts \*

```
1 import { NgModule } from '@angular/core';
2 import { Routes, RouterModule } from '@angular/router';
3
4 const routes: Routes = [
5   { path: '', redirectTo: 'home', pathMatch: 'full' },
6   { path: 'home', loadChildren: './home/home.module#HomePageModule' },
7 ];
8
9 @NgModule({
10   imports: [RouterModule.forRoot(routes)],
11   exports: [RouterModule]
12 })
13 export class AppRoutingModule { }
```

Fichier de routes permet la configuration de la navigation entre les différentes pages

# Structure du projet: **app-routing.module.ts**

- Une route définit simplement un chemin d'URL et un composant pour afficher son contenu. Vous utiliserez fréquemment trois types principaux de routes, (1) eager-loaded (2) lazy-loaded (Ionic Pages), and (3) redirects.
  - Eager. Routes qui pointe vers un seul composant.
  - Lazy. Routes qui pointe vers un module.
  - Redirect. Routes qui redirige vers une autre route.

```
const routes: Routes = [
  // Regular Route
  { path: 'eager', component: HomePage },
  // Lazy Loaded Route (Page)
  { path: 'lazy', loadChildren: './home/home.module#HomePageModule' },
  // Redirect
  { path: 'here', redirectTo: 'there', pathMatch: 'full' }
];
```

# Ajouter une nouvelle page: *ionic g*

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

PS C:\Users\OSall\Desktop\AppIonic\monProjet> ionic g
? What would you like to generate? (Use arrow keys)
? What would you like to generate? page
? Name/path of page: secondepage
> ng generate page secondepage
CREATE src/app/secondepage/secondepage.module.ts (568 bytes)
CREATE src/app/secondepage/secondepage.page.html (138 bytes)
CREATE src/app/secondepage/secondepage.page.spec.ts (726 bytes)
CREATE src/app/secondepage/secondepage.page.ts (276 bytes)
CREATE src/app/secondepage/secondepage.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (471 bytes)
[OK] Generated page!
PS C:\Users\OSall\Desktop\AppIonic\monProjet>
```

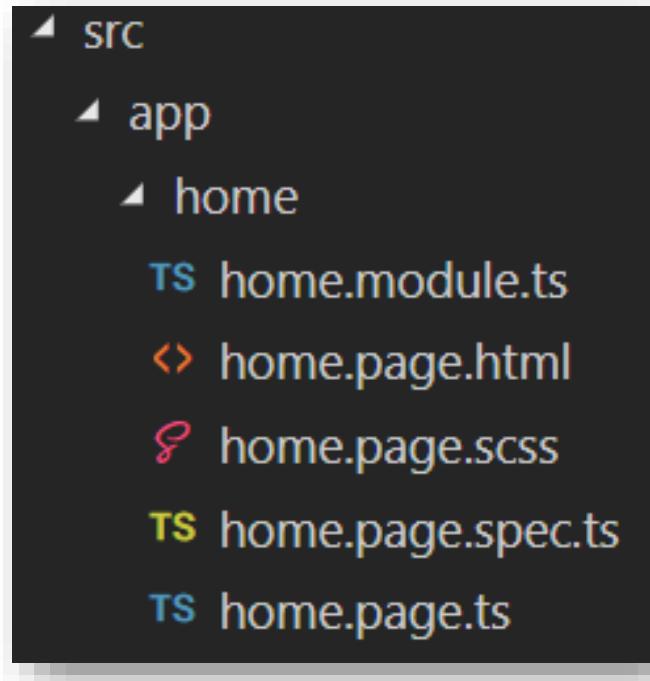
Fais le mapping entre seconde page et le composant seconde page dans le fichier secondepage.module.ts

```
TS app-routing.module.ts ×
  1 import { NgModule } from '@angular/core';
  2 import { Routes, RouterModule } from '@angular/router';
  3
  4 const routes: Routes = [
  5   { path: '', redirectTo: 'home', pathMatch: 'full' },
  6   { path: 'home', loadChildren: './home/home.module#HomePageModule' },
  7   { path: 'secondepage', loadChildren: './secondepage/secondepage.module#Sec
  8 ];
  9
 10 @NgModule({
 11   imports: [RouterModule.forRoot(routes)],
 12   exports: [RouterModule]
 13 })
 14 export class AppRoutingModule { }
```

# Ajouter une nouvelle page

- Chaque composant se compose principalement des éléments suivants:
  - Fichier de configuration des routes dans des modules autre que le module racine utilisant la méthode `forchild` ([home.module.ts](#)):  
Exemple : `RouterModule.forChild( [ {  
    path: 'ROUTE_VERS_COMPOSANT',  
    component: COMPOSANT_A_CHARGER }  
])`
    - HTML Template : représentant sa vue([home.page.html](#))
    - Une classe typescript représentant sa logique métier([home.page.ts](#))
    - Une feuille de style CSS([home.page.scss](#))
    - Un fichier spec sont des tests unitaires([home.page.spec.ts](#))
  - Les composants sont facile à mettre à jour et à échanger entre les différentes parties des applications.

Du point de vue angulaire, mais les pages et les composants ont une signification différente en Ionic. Les deux ne sont que des composants, mais une page est un composant qui agira comme une vue complète (elle peut avoir des composants imbriqués); nous considérons les pages ioniques comme un concept autonome. Un composant fera simplement partie d'un plus gros composant la plupart du temps dans les applications angulaires



# Ouvrir home.page.html

The screenshot shows two windows side-by-side. On the left is Visual Studio Code with the title "home.page.html - monProjet - Visual Studio Code". The code editor displays the following HTML:

```
<ion-header>
<ion-toolbar>
<ion-title>
  Ionic Blank
</ion-title>
</ion-toolbar>
</ion-header>

<ion-content padding>
<ion-button href="/secondepage">
  Seconde Page
</ion-button>
</ion-content>
```

A callout box points to the button element with the text "Ajoutons un bouton permettant de charger la seconde page". Below the code editor is a terminal window showing the command "ionic g" being run in a Windows PowerShell.

On the right is an Ionic Lab browser window titled "Ionic Lab" with the URL "localhost:8200". It displays a mobile application interface with the title "Ionic Blank" and a blue button labeled "SECONDE PAGE". A callout box points to this button with the text "Va aller dans secondepage.module.ts et charger le composant SecondePage".

• secondepage.page.html - monProjet - Visual Studio Code

File Edit Selection View Go Debug Terminal Help

EXPLORER ie.page.html secondepage.page.html app-routing.module.ts home.module.ts

OPEN EDITORS 1 UNSAVED

home.page.html src\app\... secondepage.page.html ... app-routing.module.ts ... home.module.ts ... home.page.spec.ts ... secondepage.module.ts ...

MONPROJET

src app home secondepage

home.module.ts home.page.html home.page.scss home.page.spec.ts home.page.ts secondepage.module.ts secondepage.page.html secondepage.page.scss secondepage.page.sp... secondepage.page.ts app-routing.module.ts app.component.html app.component.spec.ts app.components.ts app.module.ts assets environments

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: powershell

```
PS C:\Users\OSall\Desktop\AppIonic\monProjet> ionic g
? What would you like to generate? (Use arrow keys)
? What would you like to generate? page
? Name/path of page: secondepage
> ng generate page secondepage
CREATE src/app/secondepage/secondepage.module.ts (568 bytes)
(568 bytes) (138 bytes)
CREATE src/app/secondepage/secondepage.page.html (138 bytes)
CREATE src/app/secondepage/secondepage.page.spec.ts (726 bytes)
CREATE src/app/secondepage/secondepage.page.ts (276 bytes)
CREATE src/app/secondepage/secondepage.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (471 bytes)
[OK] Generated page!
```

Ionic Lab localhost:8200

Applications Bookmarks Connexion

Open fullscreen Platforms

secondepage

ACCUEIL

monProjet - 0.0.1

Twitter Documentation Forum GitHub

Autre possibilité:

[routerLink]=["['/path', routeParam]"]>

```
<ion-button expand="full" [routerLink]=["'/secondepage', value]" routerDirection="forward">
  Push with router Link
</ion-button>
```

```
<a routerLink="/path-name-to-page">
  Aller à une autre page
</a>
```

# Autre possibilité: Naviguer par Programmation avec Angular Router

- Vous pouvez également naviguer dynamiquement à partir d'un composant ou d'un service. Un cas d'utilisation courant consiste à attendre que quelque chose d'async se termine, puis à envoyer l'utilisateur à une nouvelle page.
- Angular fournit un service de routeur que nous pouvons injecter dans les composants en le déclarant dans le constructeur.

TS home.page.ts ×

```
1 import { Component } from '@angular/core';
2 import { Router } from '@angular/router';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   constructor(private router: Router) { }
11
12   go() {
13     this.router.navigateByUrl('/secondepage');
14   }
15 }
```

# Autre possibilité: Naviguer avec des Urls dynamiques

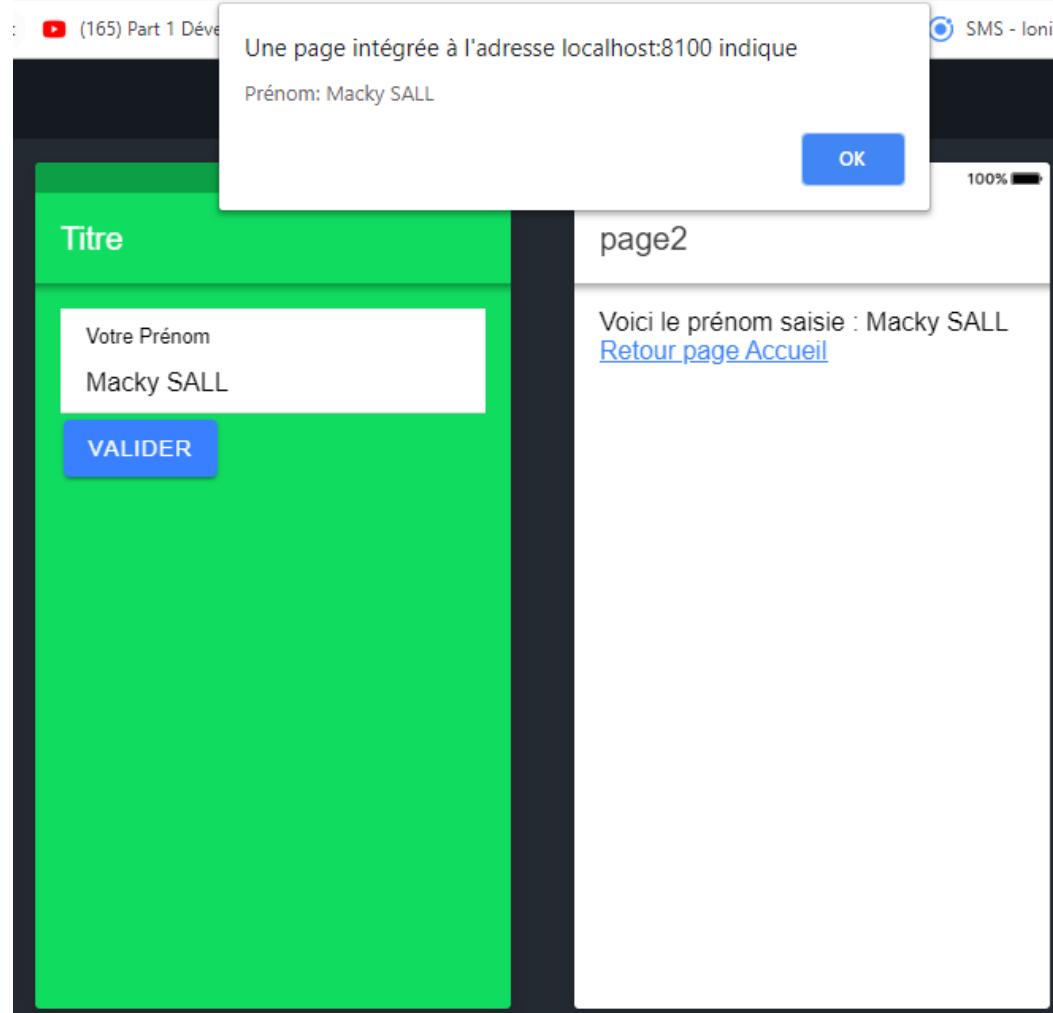
- Dans de nombreux cas, vous devez accéder à un chemin qui ressemble à **/items/:id** et vous ne connaissez pas la valeur de l'ID à l'avance. Configurons un itinéraire capable de gérer les paramètres dynamiques. Vous pouvez créer un segment d'URL dynamique en le prefixant avec **:**, technique utile pour le partage de données entre des composants ou des pages.

```
const routes: Routes = [  
  // Regular Route  
  { path: 'items/:id', component: MyComponent },  
];
```

- Tout modèle de chaîne valide peut être utilisé pour le deuxième segment d'URL. Les deux rendront le même composant, mais passeront un ID différent à travers les paramètres.

```
<ion-button href="/items/abc">ABC</ion-button>  
<ion-button href="/items/xyz">XYZ</ion-button>
```

# Naviguer d'une page à l'autre avec transfert de données: **NavController** et **ActivatedRoute**

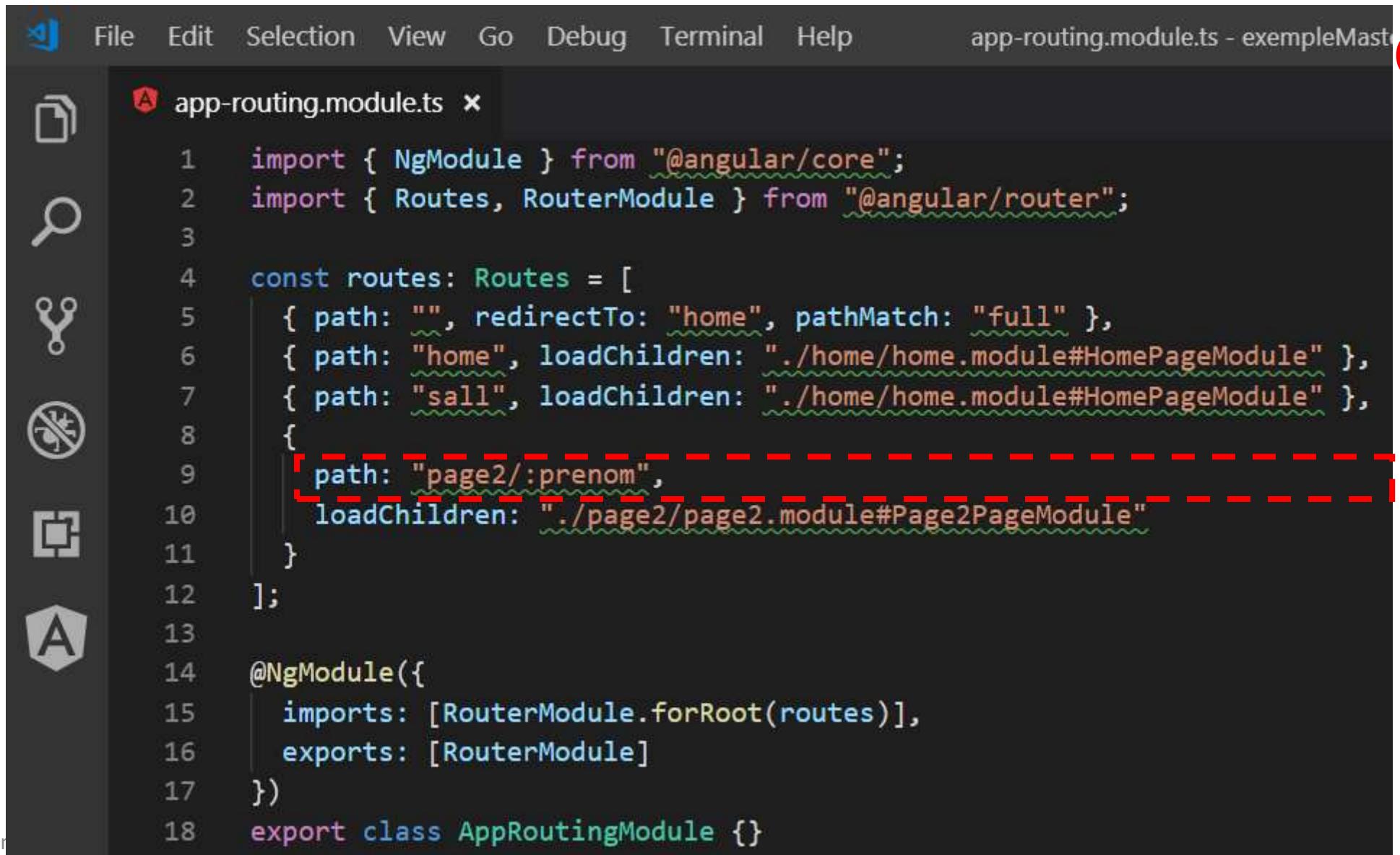


# Naviguer d'une page à l'autre avec transfert de données: **NavController** et **ActivatedRoute**

'@ionic/angular' '@ionic/angular'

- **NavController** est la classe de base pour les composants de contrôleur de navigation tels que Nav et Tab.
- **NavController** est utilisé pour naviguer entre les pages d'une application. Au niveau de base, un contrôleur de navigation est un tableau de pages représentant un historique particulier (d'un onglet par exemple). Ce tableau peut être manipulé pour naviguer dans une application en poussant et en sautant des pages ou en les insérant et en les supprimant à des emplacements arbitraires de l'historique.

# Naviguer d'une page à l'autre avec transfert de



```
File Edit Selection View Go Debug Terminal Help app-routing.module.ts - exempleMaster
A app-routing.module.ts ×

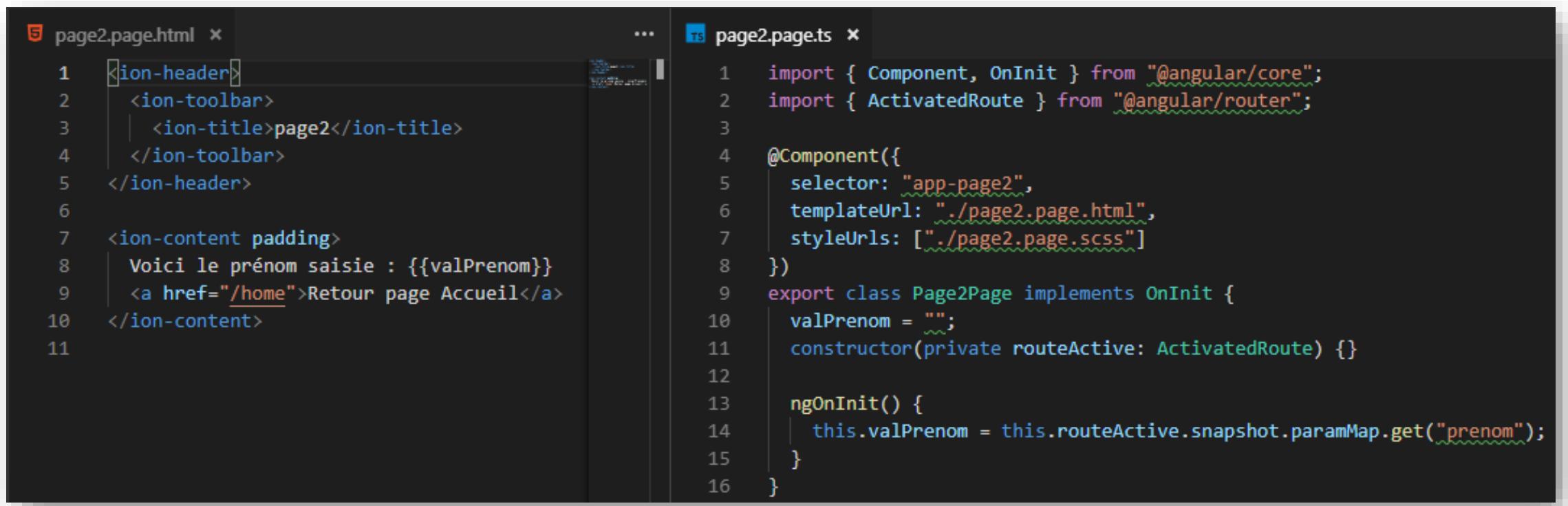
1 import { NgModule } from "@angular/core";
2 import { Routes, RouterModule } from "@angular/router";
3
4 const routes: Routes = [
5   { path: "", redirectTo: "home", pathMatch: "full" },
6   { path: "home", loadChildren: "./home/home.module#HomePageModule" },
7   { path: "sall", loadChildren: "./home/home.module#HomePageModule" },
8   {
9     path: "page2/:prenom",
10    loadChildren: "./page2/page2.module#Page2PageModule"
11  }
12];
13
14 @NgModule({
15   imports: [RouterModule.forRoot(routes)],
16   exports: [RouterModule]
17 })
18 export class AppRoutingModule {}
```

# Naviguer d'une page à l'autre avec transfert de données: NavController et ActivatedRoute

The screenshot shows a code editor with two files open:

- home.page.ts**: A TypeScript file containing the component definition for the home page. It includes a constructor injecting a `NavController`, an `onClick` event handler that alerts the current prenom and then navigates forward to a new page with the same prenom as a query parameter, and a class variable `prenom` set to "Moussa Lo".
- home.page.html**: An HTML file defining the UI for the home page. It features an ion-header with a title "Titre", an ion-content section with a floating label and input field for "Votre Prénom", and an ion-button labeled "Valider".

# Naviguer d'une page à l'autre avec transfert de données: NavController et ActivatedRoute



The image shows a code editor with two files side-by-side:

- page2.page.html**: An Ionic template file containing:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>page2</ion-title>
4   </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8   Voici le prénom saisis : {{valPrenom}}
9   <a href="/home">Retour page Accueil</a>
10 </ion-content>
11
```
- page2.page.ts**: An Angular component file containing:

```
1 import { Component, OnInit } from "@angular/core";
2 import { ActivatedRoute } from "@angular/router";
3
4 @Component({
5   selector: "app-page2",
6   templateUrl: "./page2.page.html",
7   styleUrls: ["./page2.page.scss"]
8 })
9 export class Page2Page implements OnInit {
10   valPrenom = "";
11   constructor(private routeActive: ActivatedRoute) {}
12
13   ngOnInit() {
14     this.valPrenom = this.routeActive.snapshot.paramMap.get("prenom");
15   }
16 }
```

# Pour changer le nom de l'application

- Aller dans le fichier **config.xml**(Si le fichier est absent de l'arborescence exécuter la commande: **ionic integrations enable cordova -add**)
- Ensuite mettez le nom choisi entre les tags **<name> ? </name>**

# Pour changer l'icône de l'application

- Créez une icon.png avec comme taille size 1024x1024 pixels et **sans transparence**. Cela peut être fait en utilisant “paint.exe” et lors de l'enregistrement, il vous sera demandé de supprimer la transparence.
- Créez un fichier splash.png de 2748x2748 pixels **sans transparence**. Pour cela, vous pouvez utiliser “paint.exe”. Lors de la sauvegarde, il vous sera demandé de supprimer la transparence.
- Copiez ces images dans le dossier des ressources du projet Ionic.

```
# config.xml
<widget>
...
<icon src="/path/to/generic/icon" /> # generic icon
<platform name="android">
  <icon src="/path/to/android/icon" /> # android specific icon
</platform>
...
</widget>
```

# Pour créer une application et publier dans playstore

<https://github.com/hughred22/YouTube-Video-Listing-Ionic-Mobile-App/wiki/Preparing-for-Release-and-Publishing-Your-Ionic-App>

1. Ajoutez le support de la plateforme Android à votre application si vous ne l'avez pas déjà fait. Dans le dossier racine de votre application, tapez terminal : **ionic platform add android**
2. Créez votre clé numérique signée **\$ keytool -genkey -v -keystore my-release-key.keystore -alias alias\_name -keyalg RSA -keysize 2048 -validity 10000** Suivez les instructions à l'écran pour saisir votre mot de passe et vos informations. Veuillez enregistrer cette clé dans un endroit sûr. Vous en aurez besoin à nouveau lorsque vous signerez une nouvelle version et publierez une mise à jour.
3. Construire une version de votre application **\$ ionic cordova build --release android** Il générera un fichier **android-release-unsigned.apk** sous le chemin d'accès à votre platform/android/build/output/apk. Copiez le fichier apk dans le dossier racine de votre application afin que vous n'ayez pas besoin de taper le chemin..

# Pour créer une application et publier dans playstore

<https://github.com/hughred22/YouTube-Video-Listing-Ionic-Mobile-App/wiki/Preparing-for-Release-and-Publishing-Your-Ionic-App>

1. Signez votre release build apk. `$ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore android-release-unsigned.apk alias_name`
2. Alignez votre construction avec l'outil zipalign pour l'outil de construction Android. `$ [Your Android SDK path]/build-tools/[Newest version number]/zipalign -v 4 android-release-unsigned.apk myapp-signed.apk` Si vous continuez à obtenir une erreur de commande non trouvée, vous pouvez également naviguer dans votre dossier Android sdk, trouver l'outil zipalign sous build-tools, numéro de version et faire glisser le dossier dans votre terminal pour obtenir le chemin absolu.

# Pour créer une application et publier dans playstore

<https://github.com/hughred22/YouTube-Video-Listing-Ionic-Mobile-App/wiki/Preparing-for-Release-and-Publishing-Your-Ionic-App>

- Myapp-signed.apk est le fichier apk final signé que vous devez télécharger sur Google Play Store ou sur tout autre réseau de distribution Android.
- **Publiez votre application sur Google Play Store.** Maintenant que notre version APK est prête pour le Google Play Store, nous pouvons créer une liste Play Store et télécharger notre APK. Pour commencer, vous devrez visiter the [Google Play Store Developer Console](#) et créez un nouveau compte développeur. Il vous en coûtera 25 \$ une fois.

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles hybrides

## Ionic - Introduction à TypeScript



# Introduction à TypeScript



Plus d'images

## TypeScript

Langage de programmation

TypeScript est un langage de programmation libre et open source développé par Microsoft qui a pour but d'améliorer et de sécuriser la production de code JavaScript. C'est un sur-ensemble de JavaScript.  
[Wikipédia](#)

Conçu Par : [Microsoft](#)

Extension de fichier : ts

Date de première version : 9 février 2012

Dernière version : 3.2.2 (29 novembre 2018)

Typage : dynamique, faible, fort optionnel, statique optionnel

Paradigme : Multi-paradigme

# Introduction à TypeScript

- **TypeScript** est un langage de programmation libre et open source développé par Microsoft qui a pour but d'améliorer et de sécuriser la production de code JavaScript.
- C'est un sur-ensemble de JavaScript (c'est-à-dire que tout code JavaScript correct peut être utilisé avec TypeScript).
- Le code TypeScript est transcompilé en JavaScript, pouvant ainsi être interprété par n'importe quel navigateur web
- ou moteur JavaScript.
- Il a été cocréé par Anders Hejlsberg, principal inventeur de C#
- TypeScript permet un typage statique optionnel des variables et des fonctions, la création de classes et d'interfaces, l'import de modules, tout en conservant l'approche non- contraignante de JavaScript.
- Il supporte la spécification ECMAScript 6.

# Introduction à TypeScript

- Types de base
- fonction fléchée (arrow function en anglais)
- Interfaces
- Classes
- Constructeurs
- Modificateurs
- Propriétés
- Modules

# Installation

```
C:\Users\Developpeur>npm install -g typescript
C:\Users\Developpeur\AppData\Roaming\npm\tsserver -> C:\Users\Developpeur\AppData\Roaming\npm\node_modules\typescript\bin\tsserver
C:\Users\Developpeur\AppData\Roaming\npm\tsc -> C:\Users\Developpeur\AppData\Roaming\npm\node_modules\typescript\bin\tsc
+ typescript@3.3.333
added 1 package from 1 contributor in 13.168s

C:\Users\Developpeur>cd Desktop

C:\Users\Developpeur\Desktop>cd AppIonic

C:\Users\Developpeur\Desktop\AppIonic>code exemple.ts
```

# Introduction à TypeScript: Messages de log

- L'instruction qui permet d'afficher des messages de log en TypeScript est :

```
console.log('Hello, world!');
```



exemple.ts x

1 console.log('Bonjour Tout le monde');



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: powershell



```
PS C:\Users\Developpeur> cd .\Desktop\AppIonic
PS C:\Users\Developpeur\Desktop\AppIonic> tsc exemple.ts
PS C:\Users\Developpeur\Desktop\AppIonic> dir
```



Répertoire : C:\Users\Developpeur\Desktop\AppIonic

Mode	LastWriteTime	Length	Name
----	-----	-----	
d----	08/03/2019	09:48	exemple1
d----	08/03/2019	10:18	exemple2
d----	08/03/2019	08:20	td1
d----	27/02/2019	16:23	typeBlank
-a---	12/03/2019	07:48	39 exemple.js
-a---	12/03/2019	07:40	37 exemple.ts

```
PS C:\Users\Developpeur\Desktop\AppIonic> node exemple.js
Bonjour Tout le monde
PS C:\Users\Developpeur\Desktop\AppIonic>
```

# Introduction à TypeScript: déclaration des variables

TypeScript suit les mêmes règles que JavaScript pour les déclarations de variable. Les variables peuvent être déclarées en utilisant: var, let et const.

- Déclaration avec **var** : Les variables dans TypeScript peuvent être déclarées à l'aide du mot clé **var**, comme dans JavaScript. Les règles de détermination de la portée restent les mêmes qu'en JavaScript.
- Déclaration avec **let** et **const**: Pour résoudre les problèmes liés aux déclarations **var**, ES6 a introduit deux nouveaux types de déclarations de variable en JavaScript, à l'aide des mots-clés **let** et **const**. TypeScript, étant un sur-ensemble de JavaScript, prend également en charge ces nouveaux types de déclarations de variables.

```
function uneFonction() {  
    for (var index = 0; index < 10; index++) {  
        console.log(index);  
    }  
    console.log("Last valeur de index: "+index);  
}
```

```
function uneFonction() {  
    for (let index = 0; index < 10; index) {  
        console.log(index);  
    }  
    console.log("Last valeur de index: "+index);  
}
```

Contrairement aux variables déclarées avec **var**, les variables déclarées avec **let** ont une portée de bloc. Cela signifie que la portée des variables **let** est limitée à leur bloc conteneur, par ex. fonction, bloc **if else** ou bloc de la boucle.

# Introduction à TypeScript: les types de base

Pour écrire des programmes, nous devons travailler avec certaines des unités de données les plus simples: nombres, chaînes, structures, valeurs booléennes, etc. En TypeScript, les types de base sont les mêmes que ceux en JavaScript, avec un type d'énumération pratique ajouté pour faciliter les choses.

- **Typage** : Le fait de différencier le type d'une variable Dans un langage dit typé, on ne peut enregistrer qu'un type de valeur dans une variable. Typage en TS :

```
let variable: type;  
const unNombre: number = 12;
```

- **Boolean** : Le type de données le plus élémentaire est la valeur simple **true/ false**, que JavaScript et TypeScript appellent une valeur booléenne. Exemple: `let bool: boolean = false;`

# Introduction à TypeScript: les types de base

- **Number** : Comme en JavaScript, tous les nombres en TypeScript sont de type float. Le type number en TypeScript supporte aussi les hexadecimal, les binaires, les octal :

```
let valDecimal: number = 6;
let valHex: number = 0xf00d;
let valBinaire: number = 0b1010;
let valOctale: number = 0o744;
```

- **String** : Un autre type ultra commun, le type String. Il est utilisé dès que vous voulez travailler avec des données textes. En TypeScript, il est possible d'utiliser les double quotes ou les simples quotes pour définir une variable de type String.

```
let couleur: string = "bleue";
couleur = 'rouge';
```

# Introduction à TypeScript: les types de base

```
ts home.page.ts ✘

1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage {
9 }
10
11 let nomComplet: string = "Macky SALL";
12 let age: number = 57;
13 let phrase: string = 'Hello, son nom est ' + nomComplet + '. Il sera agé de ' + (age + 1) + ' ans lannée prochaine.';
14
15 document.body.innerHTML = "<h1>Une autre manière d'afficher le contenu :</h1>" + phrase;
```

# Introduction à TypeScript: les types de base

home.page.ts

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage {
9   sentence = phrase;
10 }
11
12 let nomComplet: string = "Macky SALL";
13 let age: number = 57;
14 let phrase: string = 'Hello, son nom est ' + nomComplet
15 ||| | | | | | | | | | + '. Il sera agé de ' + (age + 1) + ' ans lannée prochaine.';
```

home.page.html

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  {{sentence}}
11 </ion-content>
```

Data Binding

# Introduction à TypeScript: les types de base

- **Array** : Comme en JavaScript, les tableaux en TypeScript sont définis par `[]`. Il y a également un autre moyen de les définir avec `Array<type>`:

```
let list: number[] = [1, 2, 3];
let list: Array<number> = [1, 2, 3];
```

```
var phrase = "<ul>";
let list: Array<number> = [1, 2, 3];
for (let index = 0; index < list.length; index++) {
    const element = list[index];
    phrase += "<ol>" + element + "</ol>"
}
phrase += "</ul>";
document.body.innerHTML = "<h1/>Voici le tableau :</h1>" + phrase;
```

Voici le tableau :

1  
2  
3

# Introduction à TypeScript: les types de base

- **Enum** : Comme dans les langages objet comme C#, une enum est un moyen de définir un ensemble de constantes nommées.

```
enum Color {Red, Green, Blue}  
let c: Color = Color.Green;
```

```
enum Color {Red = 1, Green, Blue}  
let colorName: string = Color[2];  
alert(colorName); // Displays 'Green' as it's value is 2 above
```

- **Any**: Normalement, vous avez utilisé ce type si vous développez en TypeScript. Il est possible d'utiliser ce type pour remplacer n'importe quel autre type:

```
let notSure: any = 4;  
notSure = "maybe a string instead";  
notSure = false; // okay, definitely a boolean
```

```
let liste: any[] = [1, true, "free"];  
liste[1] = 100;
```

# Introduction à TypeScript: les types de base

- **Void** : Le type void est à l'opposé du type Any. Il sert à indiquer à une fonction qu'elle ne doit rien renvoyer:

```
function warnUser(): void {  
    alert("This is my warning message");  
}
```

- Ou à une variable qui ne peut que avoir undefined ou null comme valeur:

```
let unusable: void = undefined;
```

# Introduction à TypeScript: les opérateurs === et !==

- === et !== sont les véritables opérateurs de comparaison, **toujours les utiliser!**

```
'' === '0'    // false
'' === 0      // false
0 === '0'     // false
NaN === NaN   // still weirdly false
[ ' '] === '' // false
false === undefined // false
false === null // false
null === undefined // false
```

# Introduction à TypeScript: les tableaux

- Les tableaux sont les objets pour définir des listes

```
// Creates an empty list
const list = [];
const groceries = ['milk', 'cocoa puffs'];
groceries[1] = 'kix';

// For each loop
for (let item of groceries) {
    console.log(item);
}
```

# Introduction à TypeScript: les objets

- Collection de paires clé - valeur, peut être utilisé comme une hashmap

```
const prices = {};
const scores = {
    peach: 100,
    mario: 88,
    luigi: 91
};
console.log(scores['peach']);    // 100
console.log(scores.peach);      // 100
scores.peach = 20;
console.log(scores.peach);      // 20
```

# Introduction à TypeScript: les objets - Itérer les propriétés d'un objet

- Il est possible d'itérer sur les propriétés d'un objet

```
const scores = {
    peach: 100,
    mario: 88,
    luigi: 91
};

for (let name in scores) {
    console.log(name + ':' + scores[name]);
}
```

# Introduction à TypeScript: les structures conditionnelles et itératives

- Les structures sont les mêmes qu'en JavaScript

# Introduction à TypeScript: les fonctions fléchées

- Une expression de fonction fléchée (arrow function en anglais) permet d'avoir une syntaxe plus courte que les expressions de fonction et ne possède pas ses propres valeurs pour this, arguments, super, ou new.target.
- Les fonctions fléchées sont souvent anonymes et ne sont pas destinées à être utilisées pour déclarer des méthodes.

```
let varSomme = (a: number, b: number) => {  
    return a + b;  
}
```

```
function somme(a: number, b: number): number {  
    return a + b;  
}
```

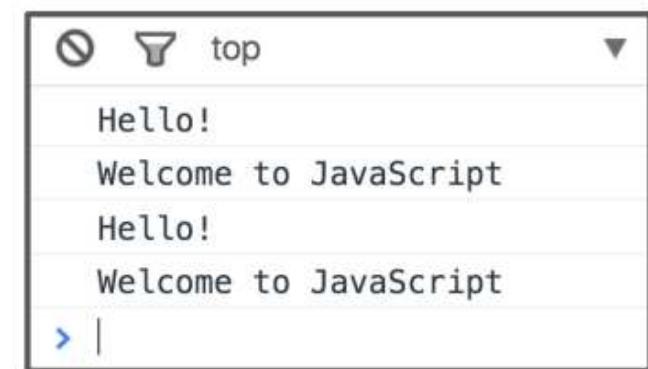
# Introduction à TypeScript: les fonctions

- La syntaxe suivante est une des manières de définir une fonction en JavaScript

```
function name() {  
    statement;  
    statement;  
    ...  
}
```

```
hello();  
hello();  
  
function hello() {  
    console.log('Hello!');  
    console.log('Welcome to JavaScript');  
}
```

Cela fonctionne car les déclarations de fonctions sont **hoisted**: déplacées au sommet du scope dans lesquelles elles sont définies  
**il faut éviter de se reposer sur ce mécanisme**



# Introduction à TypeScript: Paramètres de fonctions

```
function printMessage(message, times) {  
    for (var i = 0; i < times; i++) {  
        console.log(message);  
    }  
}
```

Les paramètres de fonctions ne sont pas déclarés à l'aide de let, const ou var

# Introduction à TypeScript: les fonctions fléchées - Syntaxe

```
([param] [, param]) => {
    instructions
}
(param1, param2, ..., param2) => expression
// équivalent à
(param1, param2, ..., param2) => {
    return expression;
}
// Parenthèses non nécessaires quand il n'y a qu'un seul argument
param => expression
// Une fonction sans paramètre peut s'écrire avec un couple de parenthèses
() => {
    instructions
}
```

# Introduction à TypeScript: les fonctions fléchées - Syntaxe

```
// Gestion des paramètres du reste et paramètres par défaut
(param1, param2, ...reste) => {
    instructions
}
(param1 = valeurDefaut1, param2, ..., paramN = valeurDefautN) => {
    instructions
}

// Gestion de la décomposition pour la liste des paramètres
let f = ([a, b] = [1, 2], { x: c } = { x: a + b }) => a + b + c;
f();
```

# Introduction à TypeScript: Interface

- L'interface est une structure qui définit le contrat dans votre application. Il définit la syntaxe des classes à suivre. Les classes dérivées d'une interface doivent suivre la structure fournie par leur interface.
- Le compilateur TypeScript ne convertit pas l'interface en JavaScript. Il utilise l'interface pour la vérification de type. Ceci est également connu comme "typage de canard" ou "sous-typage structurel".
- Une interface est définie avec le mot clé `interface` et peut inclure des propriétés et des déclarations de méthodes à l'aide d'une fonction ou d'une fonction de flèche.

# Introduction à TypeScript: les classes

```
class UneClasse {
    constructor(arg1?: type, argn?: type) {
    }

    public get arg1(): type {
        return this.arg1;
    }
    public set arg1(arg1: type) {
        this.arg1 = arg1;
    }

    public get arg2(): type {
        return this.arg2;
    }
    public set arg2(arg2: type) {
        this.arg2 = arg2;
    }
}
let uneInstance = new UneClasse(val1, val2);
```



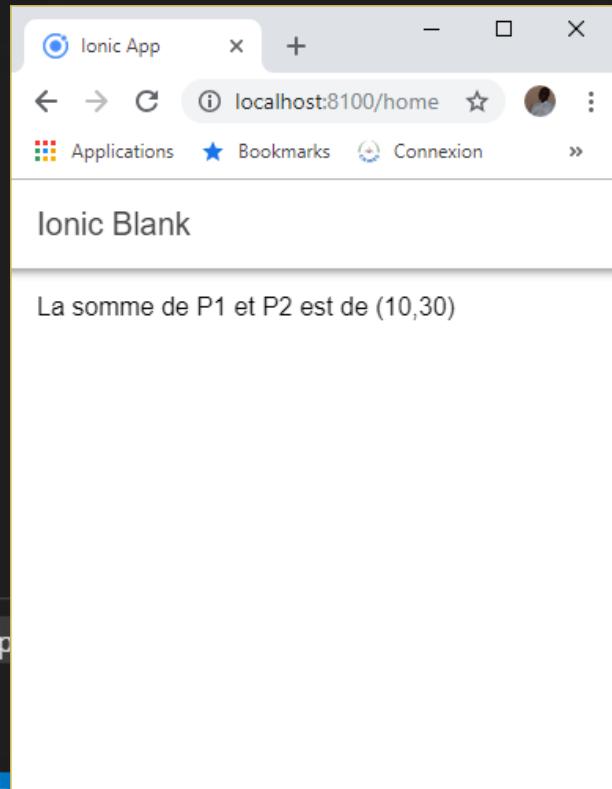
home.page.ts x

```
8  export class HomePage {
9    p1 = new Point(0, 10);
10   p2 = new Point(10, 20);
11   p3 = this.p1.add(this.p2); // {x:10,y:30}
12
13   affiche(): string {
14     return "La somme de P1 et P2 est de " + this.p3.affichePoint();
15   }
16 }
17
18 class Point {
19   x: number;
20   y: number;
21   constructor(x: number, y: number) {
22     this.x = x;
23     this.y = y;
24   }
25   add(point: Point) {
26     return new Point(this.x + point.x, this.y + point.y);
27   }
28   affichePoint() {
29     return "(" + this.x + "," + this.y + ")";
30   }
31 }
32 }
```



home.page.html x

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  {{affiche()}}
11 </ion-content>
```



PROBLÈMES

SORTIE

CONSOLE DE DÉBOGAGE

TERMINAL

PS C:\Users\0Sall\Desktop\AppIonic\projet1&gt; []

# Introduction à TypeScript: les modules

- Le code TypeScript que nous écrivons est par défaut dans la portée globale. Si nous avons plusieurs fichiers dans un projet, les variables, les fonctions, etc. écrites dans un fichier sont accessibles dans tous les autres fichiers.
- TypeScript fournit des modules et des espaces de noms afin d'empêcher la portée globale du code par défaut et également d'organiser et de gérer une base de code volumineuse.
- Les modules sont un moyen de créer une portée locale dans le fichier. Ainsi, toutes les variables, classes, fonctions, etc. déclarées dans un module ne sont pas accessibles en dehors du module. Un module peut être créé à l'aide du mot clé **export** et un module peut être utilisé dans un autre module à l'aide du mot clé **import**.

# Introduction à TypeScript: les modules

The image shows a code editor interface with two tabs: "TS main.ts" and "TS point.ts".

**point.ts:**

```
1
2 export class Point {
3     constructor(private x?: number, private y?: number) {
4     }
5
6     draw() {
7         console.log('X: ' + this.x + ', Y: ' + this.y);
8     }
9 }
```

**main.ts:**

```
1
2 import { Point } from './point';
3
4 let point = new Point(1, 2);
5 point.draw();
```

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles hybrides

## Basic building blocks of AngularJS for Ionic



The screenshot shows a code editor with a dark theme. On the left, there's a vertical toolbar with icons for file operations, search, and other developer tools. The main area displays an Ionic template file named `home.page.html`. The code uses the Ionic framework's component-based structure (`<ion-header>`, `<ion-content>`) and includes two input fields (`<ion-item>`) with labels (`<ion-label>`) and a submit button (`<button ion-button>`). A large callout box labeled "Tags Ionic: composant web" points to the `<ion-content>` tag at line 9. Another callout box labeled "Directives" points to the `[ (ngModel)]` attribute on the first input field at line 13. A third callout box points to the interpolation syntax `{{num1}}` in the string at line 21. Red dashed boxes highlight the `<ion-content>` tag, the `[ (ngModel)]` attribute, and the interpolation placeholder `num1`.

```
1  <ion-header>
2    <ion-toolbar>
3      <ion-title>
4        Ionic Blank
5      </ion-title>
6    </ion-toolbar>
7  </ion-header>
8
9  <ion-content padding>
10 <form>
11   <ion-item>
12     <ion-label>Num1</ion-label>
13     <ion-input type="number" [(ngModel)]="num1" name="num1"></ion-input>
14   </ion-item>
15   <ion-item>
16     <ion-label>Num2</ion-label>
17     <ion-input type="number" [(ngModel)]="num2" name="num2"></ion-input>
18   </ion-item>
19   <button ion-button type="submit" block>Calculer la somme</button>
20 </form>
21 La somme de {{num1}} et {{num2}} est de {{num1+num2}}
22 </ion-content>
```

# Tags Ionic: composant web

## Directives

## Expressions

```
[ng] chunk {home-home-module} home-home-module.js, home-home-module.js.map (home-home-module)
  5.87 kB [rendered]
[ng] i ｢wdm｣: Compiled successfully.
```

# Expression et data binding

Binding is just a way to connect data from the TypeScript code to the HTML. We can bind to attributes using square brackets []. For example, we might want to disable a button after it has been clicked.

- Une expression Angular est comme une expression JavaScript entourée d'accolades -  **{{expression}}** . Angular évalue l'expression spécifiée et lie les données de résultat au HTML.
- L'expression Angular peut contenir des littéraux, des opérateurs et des variables telles que l'expression JavaScript. Par exemple, une expression  **{{2+2}}**  produira le résultat 4 et sera liée à HTML.

```
<ion-content padding>
  2 + 2 = {{2 + 2}} <br />
  2 - 2 = {{2 - 2}} <br />
  2 * 2 = {{2 * 2}} <br />
  2 / 2 = {{2 / 2}}
</ion-content>
```

# La notion de data-binding

- La plupart des frameworks qui permettent de créer des applications web utilisent cette notion de data-binding. C'est le cas pour Angular. Il s'agit d'un moyen de lier la partie vue à la partie logique. En d'autres termes, grâce à cela, les éléments de votre code HTML seront liés à votre contrôleur JavaScript.

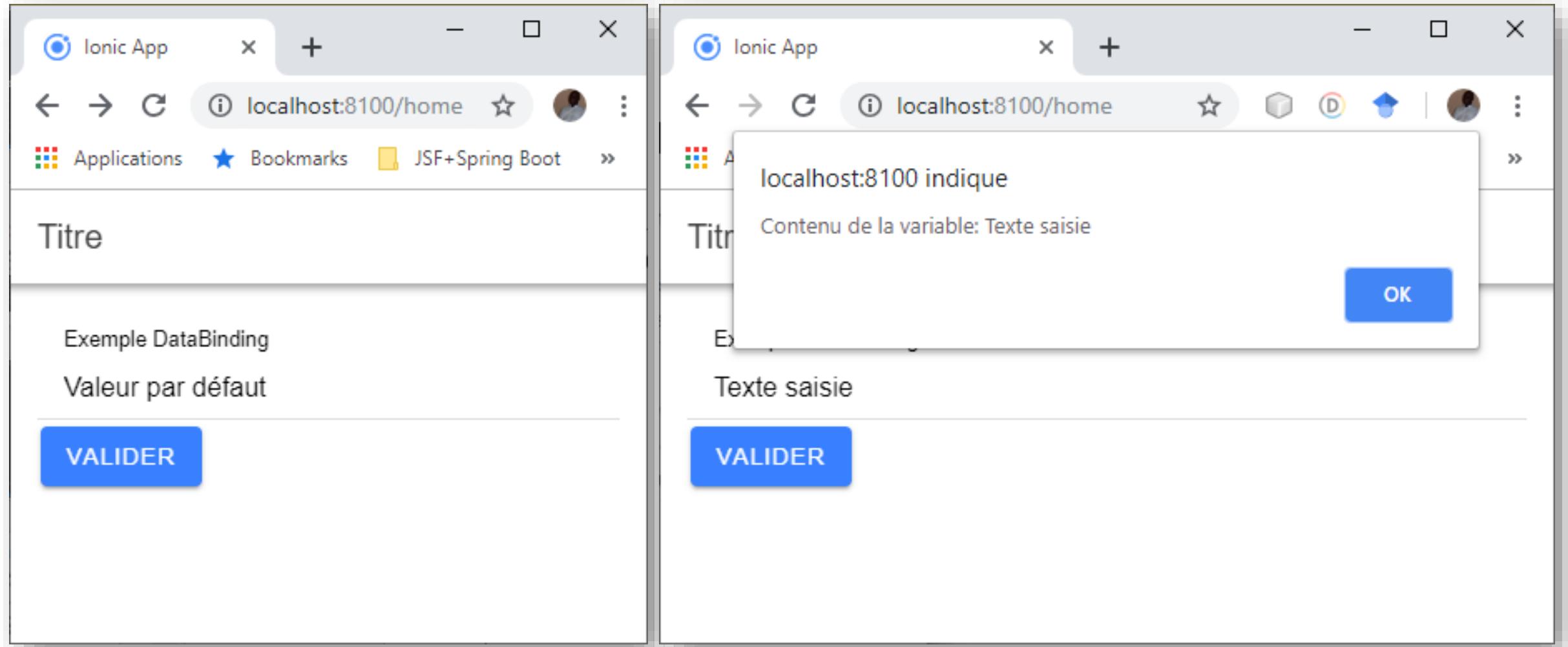


AngularJS pratique ce que l'on appelle le "two-way data binding". En d'autres termes, il s'agit d'un data-binding bidirectionnel. Ainsi, grâce à cette technique, lorsque vous ferez des actions sur votre page HTML, les variables dans votre JavaScript se mettront à jour et vice-versa : lorsque vous changerez vos variables dans votre JavaScript, les répercussions sur la vue seront immédiates.

# Les directives: `[(ngModel)]="name "`

- Directive qui lie la valeur des contrôles HTML (ion-input, ion-select, ion-textarea) aux données de l'application.
- La directive ng-model peut également:
  - Fournir la validation de type pour les données d'application (numéro, email, requis).
  - Indiquer le statut des données d'application (non valide, sale, touché, erreur).
  - Fournir des classes CSS pour les éléments HTML.
  - Lier des éléments HTML aux formulaires HTML.

# Les directives: [(ngModel)]="name "



# Les directives: [(ngModel)]="name "

The image shows a code editor with two files side-by-side:

- home.page.html**: An Ionic template file containing an ion-header, ion-toolbar, ion-title (with the text "Titre"), and an ion-content section. Inside ion-content, there is an ion-item with an ion-label (text "Exemple DataBinding") and an ion-input (type="text" with ngModel binding to maVariale). Below it is another ion-item with an ion-button (color="primary" with click event binding to onClick).
- home.page.ts**: An Angular component file defining a HomePage class with a maVariale property set to "Valeur par défaut". It contains an onClick() method that alerts the current value of maVariale.

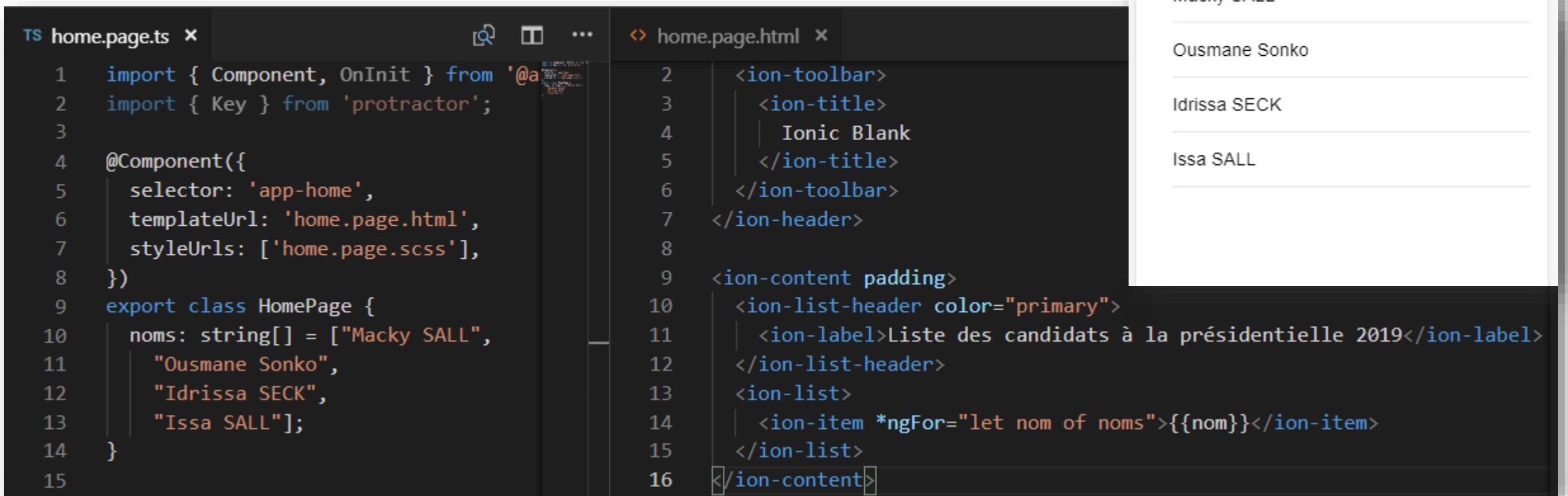
```
home.page.html
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Titre
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  <ion-item>
11    <ion-label position="floating"> Exemple DataBinding </ion-label>
12    <ion-input type="text" [(ngModel)]="maVariale" > </ion-input>
13  </ion-item>
14  <ion-button color="primary" (click)="onClick()">Valider</ion-button>
15 </ion-content>
```

```
home.page.ts
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss']
7 })
8 export class HomePage {
9   maVariale = 'Valeur par défaut';
10
11   onClick() {
12     alert('Contenu de la variable: ' + this.maVariale);
13   }
14 }
```

# Les directives: `*ngFor="let item of items"`

- Le `*` est un raccourci pour utiliser la nouvelle syntaxe de modèle angular avec la balise de modèle. Cela s'appelle aussi une directive structurelle.
- Il est utile de savoir que `*` n'est qu'un raccourci pour définir explicitement les liaisons de données sur une balise de modèle. La balise de modèle empêche le navigateur de lire ou d'exécuter le code qu'il contient.

# Les directives: \*ngFor="let item of items"



The screenshot shows a mobile application interface. At the top, there is a blue header bar with the text "Liste des candidats à la présidentielle 2019". Below the header is a white content area containing a list of names. The list starts with "Ionic Blank" and includes "Macky SALL", "Ousmane Sonko", "Idrissa SECK", and "Issa SALL".

**home.page.ts**

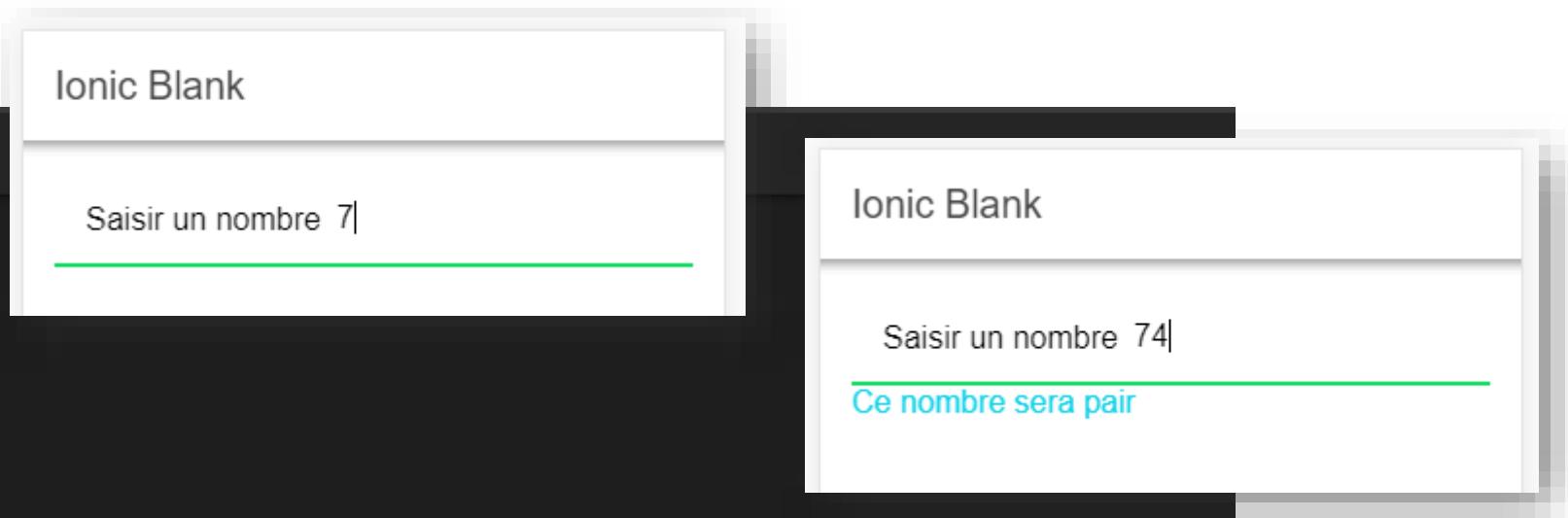
```
1 import { Component, OnInit } from '@angular/core';
2 import { Key } from 'protractor';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   noms: string[] = ["Macky SALL",
11     "Ousmane Sonko",
12     "Idrissa SECK",
13     "Issa SALL"];
14 }
15
```

**home.page.html**

```
2 <ion-toolbar>
3   <ion-title>
4     | Ionic Blank
5   </ion-title>
6 </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10 <ion-list-header color="primary">
11   <ion-label>Liste des candidats à la présidentielle 2019</ion-label>
12 </ion-list-header>
13 <ion-list>
14   <ion-item *ngFor="let nom of noms">{{nom}}</ion-item>
15 </ion-list>
16 </ion-content>
```

# Les directives: \*ngIf="expression"

<div \*ngIf="condition">Content to render when condition is true.</div>



The screenshot shows an Ionic application interface. On the left, a code editor displays the file `home.page.html` with the following content:

```
<ion-header>
  <ion-toolbar>
    <ion-title>Ionic Blank</ion-title>
  </ion-toolbar>
</ion-header>

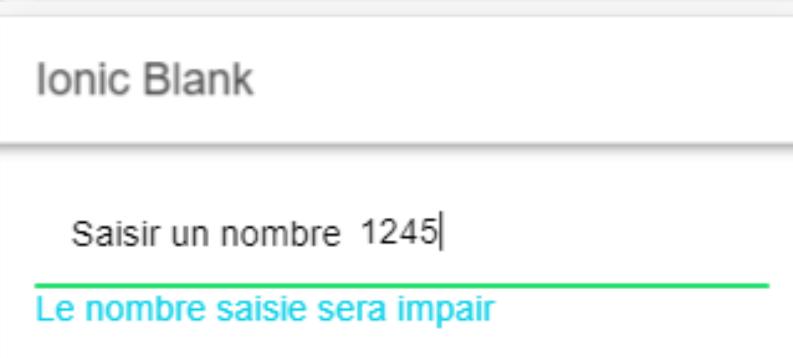
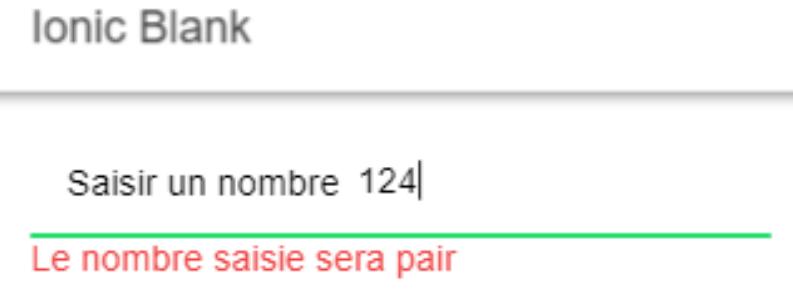
<ion-content padding>
  <ion-item>
    <ion-label color="primary">Saisir un nombre</ion-label> <br />
    <ion-input type="number" [(ngModel)]="valeur"></ion-input>
  </ion-item>
  <ion-label color="secondary" *ngIf="valeur%2==0">Ce nombre sera pair</ion-label> <br />
</ion-content>
```

Two browser windows are shown. The left window displays the input field with the value "7". The right window shows the result after entering "74", where the label "Ce nombre sera pair" is displayed below the input field.

# Les directives: \*ngIf else

```
<ng-container *ngIf="expression; else elseTemplate">
  content on true expression
</ng-container>
<ng-template #elseTemplate>
  content on false expression
</ng-template>
```

```
home.page.html x
1  <ion-header>
2    <ion-toolbar>
3      <ion-title>
4        Ionic Blank
5      </ion-title>
6    </ion-toolbar>
7  </ion-header>
8
9  <ion-content padding>
10
11  <ion-item>
12    <ion-label color="primary">Saisir un nombre</ion-label> <br />
13    <ion-input type="number" [(ngModel)]="valeur"></ion-input>
14  </ion-item>
15
16  <ng-container *ngIf="valeur%2==0; else elseTemplate">
17    <ion-text color="danger">Le nombre saisi sera pair</ion-text>
18  </ng-container>
19  <ng-template #elseTemplate>
20    <ion-text color="secondary">Le nombre saisi sera impair</ion-text>
21  </ng-template>
22
23 </ion-content>
```



# Les directives:

[ngClass] = "{'class' : true}"

- [ngClass] = "{'classname' : condition}"  
**Ou bien**
- [ngClass] = "(expr=='val')?'my-class1':'my-class2'"  
**Ou bien**
- [ngClass] = "getMaClasseCSS()"

# Les directives de gestion d'évènements

- Angular inclut certaines directives qui peuvent être utilisées pour fournir un comportement personnalisé à divers événements DOM, tels que click, dblclick, mouseenter, etc.
- Le tableau suivant répertorie les directives d'événement AngularJS.
- Voici les gestionnaires d'événements : *blur, change, click, dblclick, focus, keydown, keyup, keypress, mousedown, mouseenter, mouseleave, mousemove, mouseover, mouseup*

# Les directives de gestion d'événements: Evénement de clic

The screenshot shows the Visual Studio Code interface with two files open:

- home.page.ts**:

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage{
9   direBonjour(){
10   alert("Bonjour depuis le site VCN !!!");
11 }
12 }
```
- home.page.html**:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10 <ion-button (click)="direBonjour()">Dit Bonjour</ion-button>
11 </ion-content>
```

Below the code editor, a browser window displays the application running at `localhost:8100/home`. The page title is "Ionic Blank". A blue button labeled "DIT BONJOUR" is visible. A modal dialog box is open, showing the text "localhost:8100 indique" and "Bonjour depuis le site VCN !!!", with an "OK" button.

Fichier Modifier Sélection Afficher Accéder Déboguer Terminal Aide home.page.ts - projet1 - Visual Studio Code

home.page.ts

```
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage{
9   direBonjour(){
10   alert("Bonjour depuis le site VCN !!!");
11 }
12 }
```

home.page.html

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10 <ion-button (click)="direBonjour()">Dit Bonjour</ion-button>
11 </ion-content>
```

Ionic App

localhost:8100/home

Applications Bookmarks Connexion Développez une appli

Ionic Blank

DIT BONJOUR

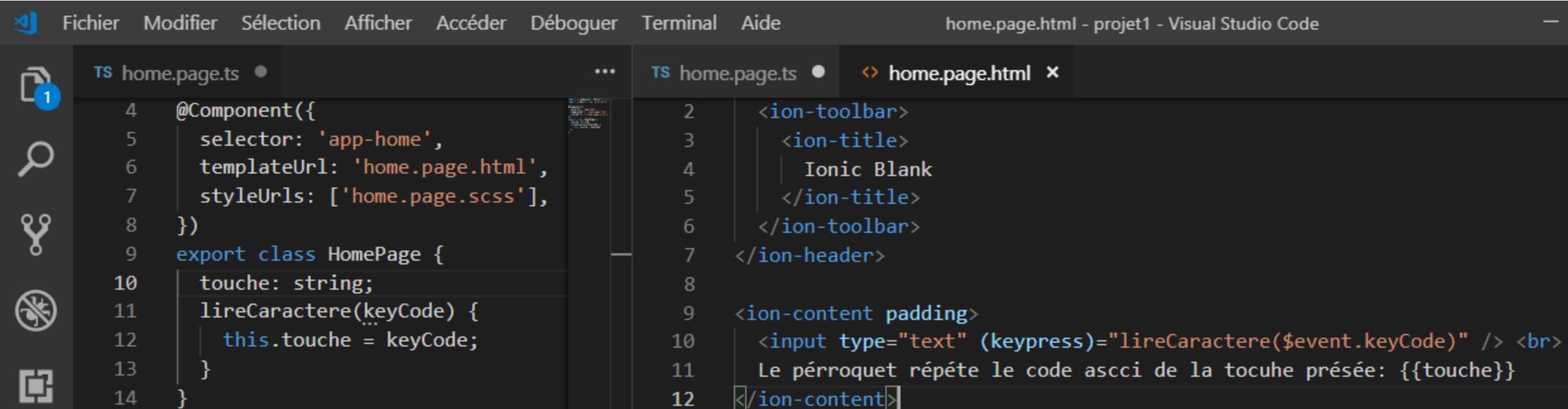
localhost:8100 indique  
Bonjour depuis le site VCN !!!

OK

Prof. Ousmane Sow

548

# Les directives de gestion d'événements: Evénements liés aux inputs - keypress



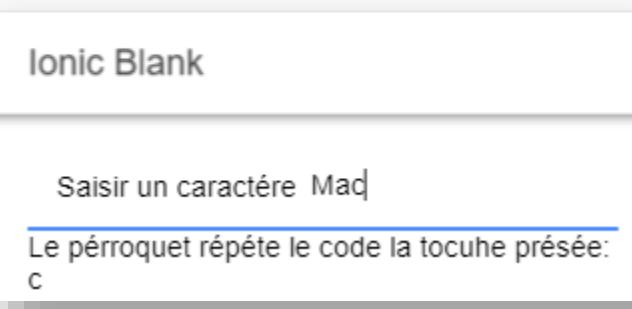
The screenshot shows the Visual Studio Code interface with two files open:

- home.page.ts**:  
A TypeScript file containing the definition of a component named `HomePage`. It includes a selector `'app-home'`, a template URL `'home.page.html'`, and a style URL `'home.page.scss'`. The `HomePage` class has a property `touche: string;` and a method `lireCaractere(keyCode) { this.touche = keyCode; }`.
- home.page.html**:  
An HTML file generated by the component. It features an `<ion-toolbar>`, an `<ion-title>` placeholder for "Ionic Blank", an `</ion-toolbar>`, an `</ion-header>`, an `<ion-content padding>` section, an `<input type="text" (keypress)="lireCaractere($event.keyCode)" />` input field, and a descriptive text: "Le pérroquet répète le code ascii de la touche pressée: {{touche}}".

# Les directives de gestion d'événements: Evénements liés aux inputs: keyup

```
TS home.page.ts x
4  @Component({
5    selector: 'app-home',
6    templateUrl: 'home.page.html',
7    styleUrls: ['home.page.scss'],
8  })
9  export class HomePage {
10    touche: string;
11    lireCaractere(event: KeyboardEvent) {
12      this.touche = event.key;
13      console.log(event);
14    }
15  }
16
17
18
```

```
TS home.page.ts x
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7   </ion-header>
8
9   <ion-content padding>
10  <ion-item>
11    <ion-label>Saisir un caractère</ion-label>
12    <ion-input type="text" (keyup)="lireCaractere($event)">
13    </ion-input> <br />
14  </ion-item>
15  Le pérroquet répète le code la touche pressée: {{touche}}
16 </ion-content>
```



```
▶ KeyboardEvent {isTrusted: true, key: "Shift", code: "ShiftLeft", location: 1, ctrlKey: false, ...} home.page.ts:13
▶ KeyboardEvent {isTrusted: true, key: "m", code: "Semicolon", location: 0, ctrlKey: false, ...} home.page.ts:13
▶ KeyboardEvent {isTrusted: true, key: "a", code: "KeyQ", location: 0, ctrlKey: false, ...} home.page.ts:13
▶ KeyboardEvent {isTrusted: true, key: "c", code: "KeyC", location: 0, ctrlKey: false, ...} home.page.ts:13
```

# Les directives de gestion d'événements:

## Evénements liés au toucher: tap, press, pan, swipe, rotate, and pinch

- Installer HammerJS

```
C:\Users\OSall\Desktop\AppIonic\exempleGestes>npm install --save hammerjs
```

<https://hammerjs.github.io/>

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# Personnaliser le thème d'une application Ionic

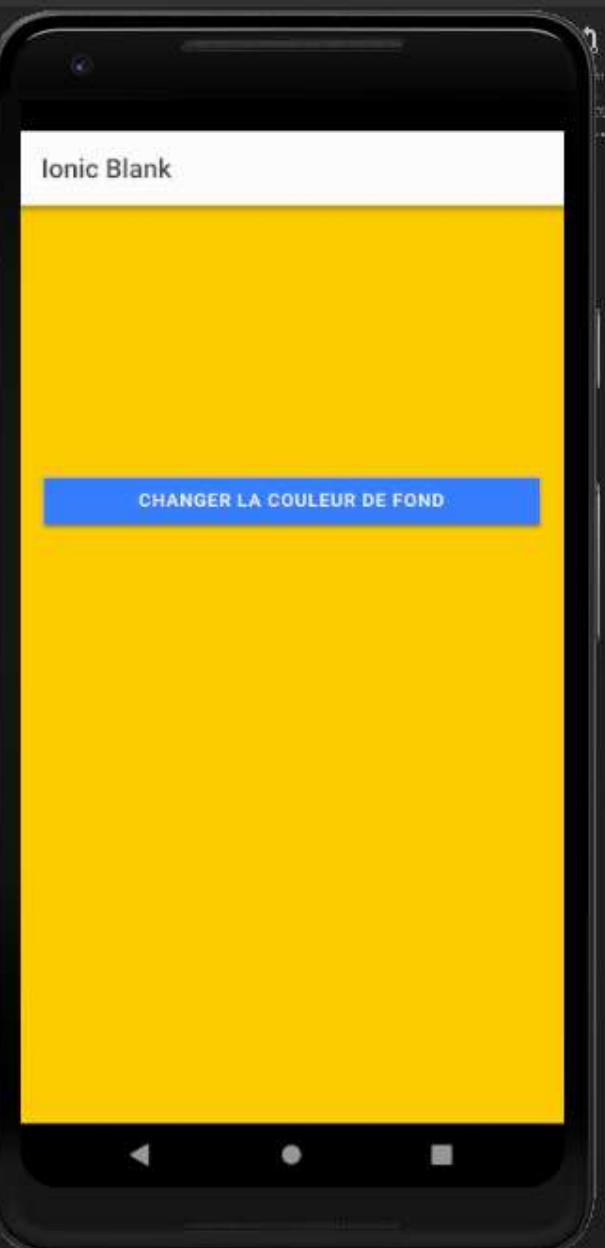


home.page.html

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding color="{{couleur}}>
10  <ion-button expand="full"
11    style="margin-top:50%"
12    [click]="changerCouleur()"
13    >Changer la couleur de fond
14  </ion-button>
15 </ion-content>
```

TS home.page.ts x

```
1 import { Component } from '@angular/core';
2 import { Color, PredefinedColors } from '@ionic/core';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   tabCouleur: Array<PredefinedColors> = ['danger',
11     'dark', 'light', 'medium', 'primary',
12     'secondary', 'success', 'tertiary', 'warning'];
13   couleur = 'warning';
14
15   changerCouleur() {
16     this.couleur =
17       this.tabCouleur[Math.floor(Math.random() * 9)];
18   }
19 }
20
```



# Personnalisation thème:

Changer le thème est aussi simple que mettre à jour la carte root dans votre fichier src / theme / variables.scss

The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with project files like 'e2e', 'node\_modules', 'src/app', 'src/assets', 'src/environments', 'src/theme', 'global.scss', 'index.html', 'karma.conf.js', 'main.ts', 'polyfills.ts', 'test.ts', 'tsconfig.app.json', 'tsconfig.spec.json', '.gitignore', and 'angular.json'. The 'variables.scss' file is selected in the Explorer and is also the active editor on the right.

The code in 'variables.scss' defines Ionic CSS Variables. It includes sections for primary, secondary, tertiary, success, warning, danger, dark, medium, and light themes, each with corresponding color definitions.

Thème	Couleur Hex
Primary	#3880ff
Secondary	#0cd1e8
Tertiary	#7044ff
Success	#10dc60
Warning	#ffce00
Danger	#f04141
Dark	#222428
Medium	#989aa2
Light	#f4f5f8

# Personnalisation thème: 9 couleurs par défaut dans Ionic 4

<https://medium.com/@paulstelzer/ionic-4-how-to-add-more-colors-and-use-them-as-color-in-buttons-and-more-7175ab4ae4e7>

- Ionic a 9 couleurs par défaut: Primary, secondary, tertiary, success, warning, danger, dark, medium et light.
- Ionic apporte un générateur de couleurs afin que vous puissiez facilement modifier ces couleurs par défaut:  
<https://ionicframework.com/docs/theming/color-generator>.
- Editez simplement les couleurs et importez-les ensuite dans votre **variable.scss** dans **src/theme**. Et tes couleurs ont été changées:



```
<ion-button color="primary">Primary</ion-button>
<ion-button class="activated" color="primary">
  Primary.activated
</ion-button>
```

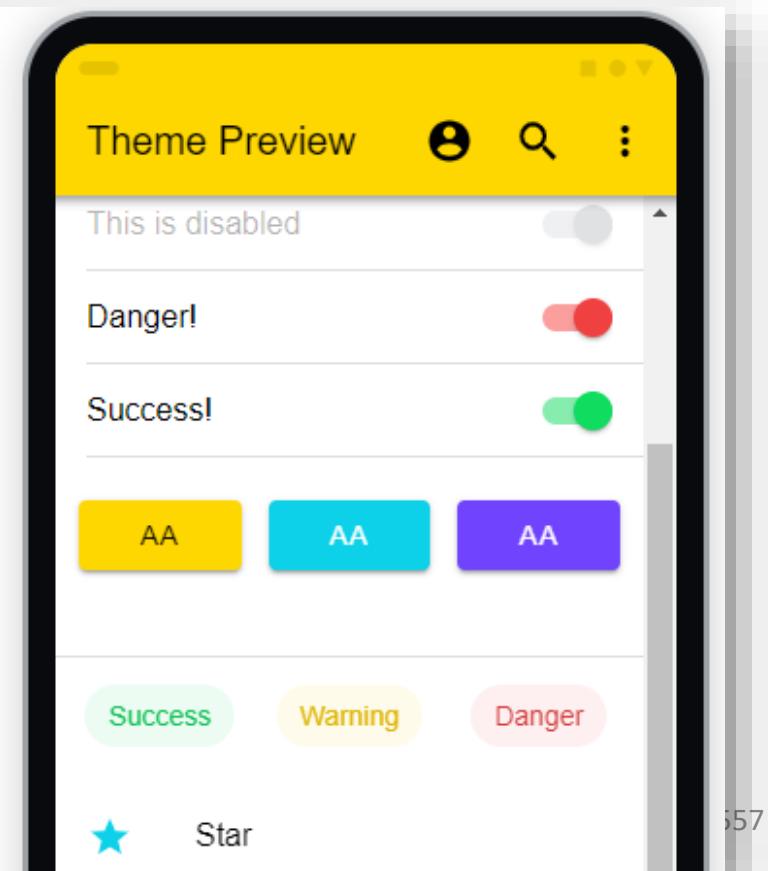
# Personnalisation thème: **Création d'une couleur personnalisée dans Ionic 4: #ffd700**

- Créez une couleur: Utiliser Ionic Color Generator accessible ici  
<https://ionicframework.com/docs/theming/color-generator>

# Color Generator

Create custom color palettes for your app's UI. Update a color's hex values, check the demo app on the right to confirm, then copy and paste the generated code directly into your Ionic project.

Primary: #ffd700  
Primary-shade: #e0bd00  
Primary-tint: #ffdb1a



The image shows the Ionic Color Generator interface on the left and a mobile application preview on the right. The generator has a sidebar with color swatches for Primary (#ffd700), Primary-shade (#e0bd00), and Primary-tint (#ffdb1a). Below the swatches is a dashed red rectangle with three input fields containing the hex codes. To the right is a mobile application preview titled "Theme Preview" showing three buttons labeled "AA" in yellow, cyan, and purple. The application also displays text "This is disabled", "Danger!", and "Success!" with corresponding toggle switches. At the bottom are buttons for "Success", "Warning", and "Danger". A "Star" button is at the bottom right of the preview.

# Personnalisation thème: Création d'une couleur personnalisée dans Ionic 4

The screenshot shows a browser window displaying the Ionic Framework documentation at <https://ionicframework.com/docs/theming/color-generator>. The page title is "Color Generator - Ionic Documentation". The main content area is titled "CSS Variables" and contains the following CSS code:

```
:root {  
  --ion-color-primary: #ffd700;  
  --ion-color-primary-rgb: 255,215,0;  
  --ion-color-primary-contrast: #000000;  
  --ion-color-primary-contrast-rgb: 0,0,0;  
  --ion-color-primary-shade: #e0bd00;  
  --ion-color-primary-tint: #ffdb1a;  
  
  --ion-color-secondary: #0cd1e8;  
  --ion-color-secondary-rgb: 12,209,232;  
  --ion-color-secondary-contrast: #ffffff;  
  --ion-color-secondary-contrast-rgb: 255,255,255;  
  --ion-color-secondary-shade: #0bb8cc;  
  --ion-color-secondary-tint: #24d6ea;
```

A red dashed box highlights the first five lines of the CSS code, specifically targeting the primary color definitions. A callout box with a red border and white background points to this highlighted area with the text "Copier la partie de la couleur primary".

# Personnalisation thème: Création d'une couleur personnalisée dans Ionic 4

The screenshot shows a code editor interface with a dark theme. The menu bar includes Fichier, Modifier, Sélection, Afficher, Accéder, Déboguer, Terminal, and Aide. The title bar says "variables.scss - projetCouleur". The left sidebar has icons for File, Search, Project, and Code. The "EXPLORATEUR" section shows "ÉDITEURS OUVERTS" with "variables.scss src\theme" and "PROJETCOULEUR" with "e2e", "node\_modules", "src" (containing "app", "assets", "environments"), "theme" (containing "variables.scss", "global.scss", "index.html", "karma.conf.js", "main.ts", "polyfills.ts", "test.ts"), and "src\theme" (containing "variables.scss"). A red dashed box highlights "variables.scss" in the sidebar and in the code editor. The code editor displays the "variables.scss" file content:

```
// Ionic Variables and Theming. For more info, please
// http://ionicframework.com/docs/theming/
//
/** Ionic CSS Variables */
:root {
    /* primary */
    --ion-color-gold: #ffd700;
    --ion-color-gold-rgb: 255, 215, 0;
    --ion-color-gold-contrast: #000000;
    --ion-color-gold-contrast-rgb: 0, 0, 0;
    --ion-color-gold-shade: #e0bd00;
    --ion-color-gold-tint: #ffdb1a;

    /* secondary */
    --ion-color-secondary: #0cd1e8;
    --ion-color-secondary-rgb: 12, 209, 232;
    --ion-color-secondary-contrast: #ffffff;
    --ion-color-secondary-contrast-rgb: 255, 255, 255;
    --ion-color-secondary-shade: #0bb8cc;
    --ion-color-secondary-tint: #24d6ea;
}
```

A red dashed box encloses the "primary" color definitions from line 6 to line 12. To the right of the editor, a callout box contains the text: "Remplacer le code css copié et les coller à la place de la couleur primary".

# Personnalisation thème: Création d'une couleur personnalisée dans Ionic 4

Fichier Modifier Sélection Afficher Accéder Déboguer Terminal Aide variables.scss - projetCouleur - Visual Studio Code

EXPLORATEUR variables.scss global.scss

EDITEURS OUVERTS variables.scss src\theme global.scss src

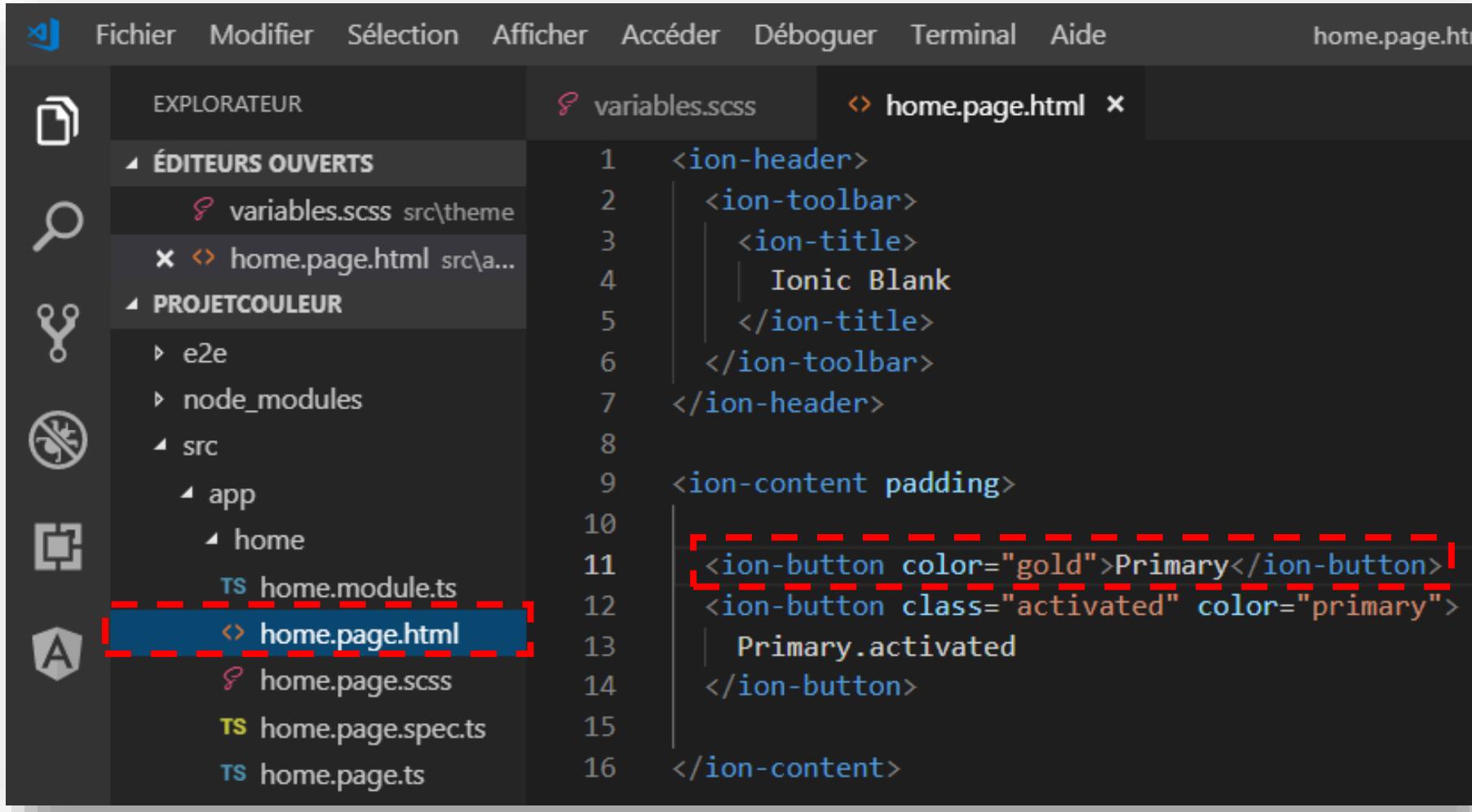
PROJETCOULEUR e2e node\_modules src app assets environments theme

Ajouter une nouvelle classe dans **variable.scss**

```
// Ionic Variables and Theming. For more info, please see:  
// http://ionicframework.com/docs/theming/  
  
.ion-color-gold {  
    --ion-color-base: var(--ion-color-gold) !important;  
    --ion-color-base-rgb: var(--ion-color-gold-rgb) !important;  
    --ion-color-contrast: var(--ion-color-gold-contrast) !important;  
    --ion-color-contrast-rgb: var(--ion-color-gold-contrast-rgb) !important;  
    --ion-color-shade: var(--ion-color-gold-shade) !important;  
    --ion-color-tint: var(--ion-color-gold-tint) !important;  
}  
/** Ionic CSS Variables **/  
  
:root {  
    /** primary **/  
    --ion-color-gold: #ffd700;  
    --ion-color-gold-rgb: 255, 215, 0;  
    --ion-color-gold-contrast: #000000;  

```

# Personnalisation thème: Création d'une couleur personnalisée dans Ionic 4



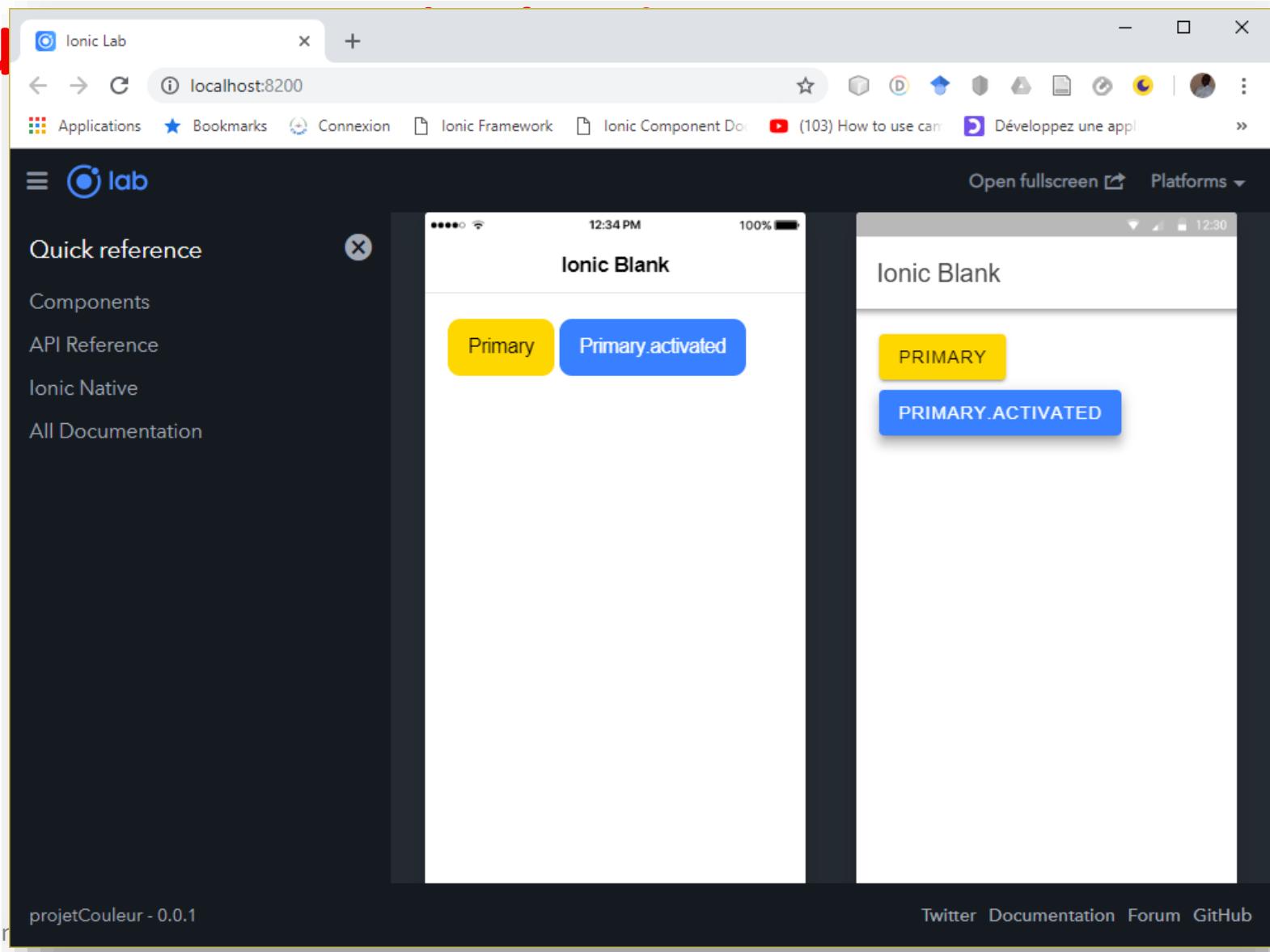
The screenshot shows a code editor interface with the following details:

- Toolbar:** Fichier, Modifier, Sélection, Afficher, Accéder, Déboguer, Terminal, Aide.
- File Explorer:** Shows the project structure:
  - ÉDITEURS OUVERTS: variables.scss, home.page.html
  - PROJETCOULEUR: e2e, node\_modules, src
    - app
      - home
        - home.module.ts
        - home.page.html
        - home.page.scss
        - home.page.spec.ts
        - home.page.ts- Code Editor:** The file `variables.scss` contains the following SCSS code:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10
11 <ion-button color="gold">Primary</ion-button>
12 <ion-button class="activated" color="primary">
13   Primary.activated
14 </ion-button>
15
16 </ion-content>
```

The line `<ion-button color="gold">Primary</ion-button>` is highlighted with a red dashed rectangle.
- Status Bar:** Shows the file name `home.page.html`.

# Personnalisation thème: Création d'une couleur



# Personnalisation thème: couleur avec la directive [ngClass] = "{'class': true}"

The screenshot shows the Visual Studio Code interface with three open files:

- home.page.scss**: Contains SCSS rules for ".bold" and ".normal" classes.
- home.page.html**: Contains an Ionic template with an `ngFor` loop and an `ngClass` directive.
- home.page.ts**: Contains the TypeScript code for the `HomePage` component.

**home.page.scss** content:

```
1 .bold{  
2   font-weight: bolder;  
3   color: blue;  
4 }  
5 .normal{  
6   font-weight: normal;  
7   color: green;  
8 }
```

**home.page.html** content:

```
1 <ion-header>  
2   <ion-toolbar color="primary">  
3     <ion-title>  
4       Ionic Blank  
5     </ion-title>  
6   </ion-toolbar>  
7 </ion-header>  
8  
9 <ion-content padding>  
10  <ion-list ng-for>  
11    <ion-item *ngFor="let nom of noms; let i=index"  
12      [ngClass]="(i % 2 == 0) ? 'bold' : 'normal'">  
13      {{i+1}} {{nom}}  
14    </ion-item>  
15  </ion-list>  
16 </ion-content>
```

**home.page.ts** content:

```
1 import { Component, OnInit } from '@angular/core';  
2 import { Key } from 'protractor';  
3  
4 @Component({  
5   selector: 'app-home',  
6   templateUrl: 'home.page.html',  
7   styleUrls: ['home.page.scss'],  
8 })  
9 export class HomePage {  
10   noms: string[] = ["Macky SALL",  
11     "Ousmane Sonko",  
12     "Idrissa SECK",  
13     "Issa SALL"];  
14 }
```

**Preview:**

Ionic Blank

- 1 Macky SALL
- 2 Ousmane Sonko
- 3 Idrissa SECK
- 4 Issa SALL

**color**

**Description**

The color to use from your application's color palette. Default options are: "primary", "secondary", "tertiary", "success", "warning", "danger", "light", "medium", and "dark". For more information on colors, see [theming](#).

563

# Personnalisation thème: utilisation d'une variable CSS dans variables.scss

- Les composants ionic sont construits avec des variables CSS pour une personnalisation aisée d'une application.
- Les variables CSS permettent de stocker une valeur dans un endroit, puis de la référencer dans plusieurs autres endroits.

**Variables globales:** le fichier src /theme/variables.scss est créé et vous pouvez remplacer les variables ioniques par défaut.

```
:root {  
  /* Set the background of the entire app */  
  --ion-background-color: #ff3700;  
  
  /* Set the font family of the entire app */  
  --ion-font-family: -apple-system, BlinkMacSystemFont, "Helvetica Neue", "Roboto", sans-serif;  
}
```

**Variables de composant:** Pour définir une variable CSS pour un composant spécifique, ajoutez la variable à l'intérieur de son sélecteur.

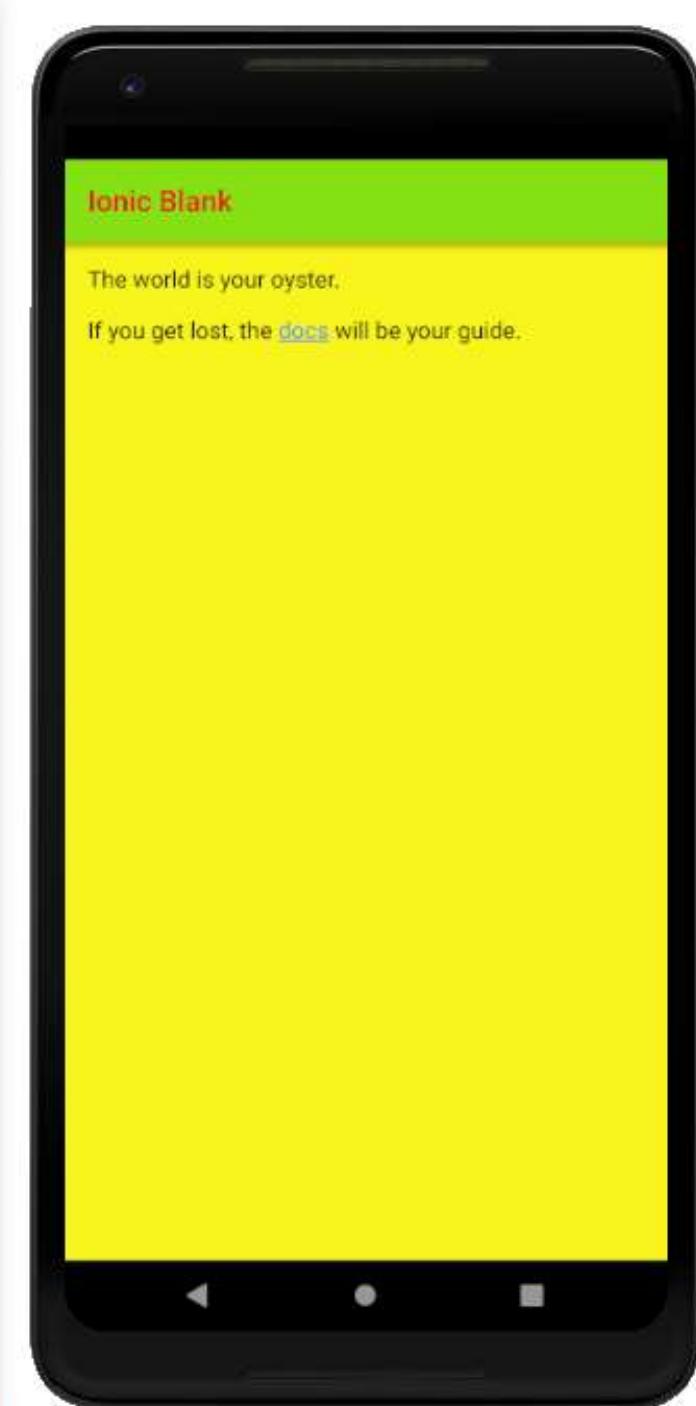
```
/* Set the color on all ion-button elements */  
ion-button {  
  | --color: #222;  
}  
  
/* Set the background on an ion-button with the .fancy-button class */  
.fancy-button {  
  | --background: #00ff00;  
}
```

Fichier Edition Sélection Affichage ... variables.scss - exempleCoule...

variables.scss x home.page.html

```
1 // Ionic Variables and Theming. For more info, please see:  
2 // http://ionicframework.com/docs/theming/  
3  
4 /** Ionic CSS Variables **/  
5 ion-toolbar{  
6   --ion-background-color: ■rgb(127, 225, 0);  
7   --color: ■#f10000;  
8 }  
9  
10 :root {  
11   --ion-background-color: ■#fcf80e;  
12  
13   /** primary **/  
14   --ion-color-primary: ■#3880ff;  
15   --ion-color-primary-rgb: 56, 128, 255;  
16   --ion-color-primary-contrast: ■#ffffff;  
17   --ion-color-primary-contrast-rgb: 255, 255, 255;  
18   --ion-color-primary-shade: ■#3171e0;  
19   --ion-color-primary-tint: ■#4c8dff;  
20  
21   /** secondary **/  
22   --ion-color-secondary: ■#0cd1e8;  
23   --ion-color-secondary-rgb: 12, 209, 232;  
24   --ion-color-secondary-contrast: ■#ffffff;  
25   --ion-color-secondary-contrast-rgb: 255, 255, 255;  
26   --ion-color-secondary-shade: ■#0bb8cc;
```

0 0 ▲ 0 AngularDoc Li 6, Col 12 Espaces : 2 UTF-8 LF SCSS 1



# Personnalisation thème: **Alignment du texte**

Attribute	Style Rule	Description
text-left	text-align: left	The inline contents are aligned to the left edge of the line box.
text-right	text-align: right	The inline contents are aligned to the right edge of the line box.
text-start	text-align: start	The same as text-left if direction is left-to-right and text-right if direction is right-to-left.
text-end	text-align: end	The same as text-right if direction is left-to-right and text-left if direction is right-to-left.
text-center	text-align: center	The inline contents are centered within the line box.
text-justify	text-align: justify	The inline contents are justified. Text should be spaced to line up its left and right edges to the left and right edges of the line box, except for the last line.
text-wrap	white-space: normal	Sequences of whitespace are collapsed. Newline characters in the source are handled as other whitespace. Breaks lines as necessary to fill line boxes.
text nowrap	white-space: nowrap	Collapses whitespace as for normal, but suppresses line breaks (text wrapping) within text.

# Personnalisation thème: Alignement du texte

```
↳ home.page.html ×  
1  <ion-header>  
2    <ion-toolbar color="primary">  
3      <ion-title>  
4        Ionic Blank  
5      </ion-title>  
6    </ion-toolbar>  
7  </ion-header>  
8  
9  <ion-content padding>  
10   <ion-list>  
11     <ion-item *ngFor="let nom of noms; let i=index" [ngClass]="(i % 2 == 0) ? 'bold' : 'normal'">  
12       {{i+1}} {{nom}}  
13     </ion-item>  
14   </ion-list>  
15   <div text-justify>La campagne électorale a démarré ce dimanche 03 janvier. Et déjà, le chef de l'  
16     commencé à lancer des piques à son opposition.  
17     «On ne doit pas accepter que des gens viennent semer la terreur. Le pouvoir, c'est Dieu qui le  
18     c'est pourquoi nous sommes venus solliciter des prières sur cette terre sainte de Serigne Touba  
19     Touba.Le candidat Macky Sall va tenir un meeting à Mbacké en début de soirée avant de faire cap  
20     Le convoi devra passer la nuit dans la ville sainte de Tivaouane.</div>  
21 </ion-content>
```

## Ionic Blank

1 Macky SALL

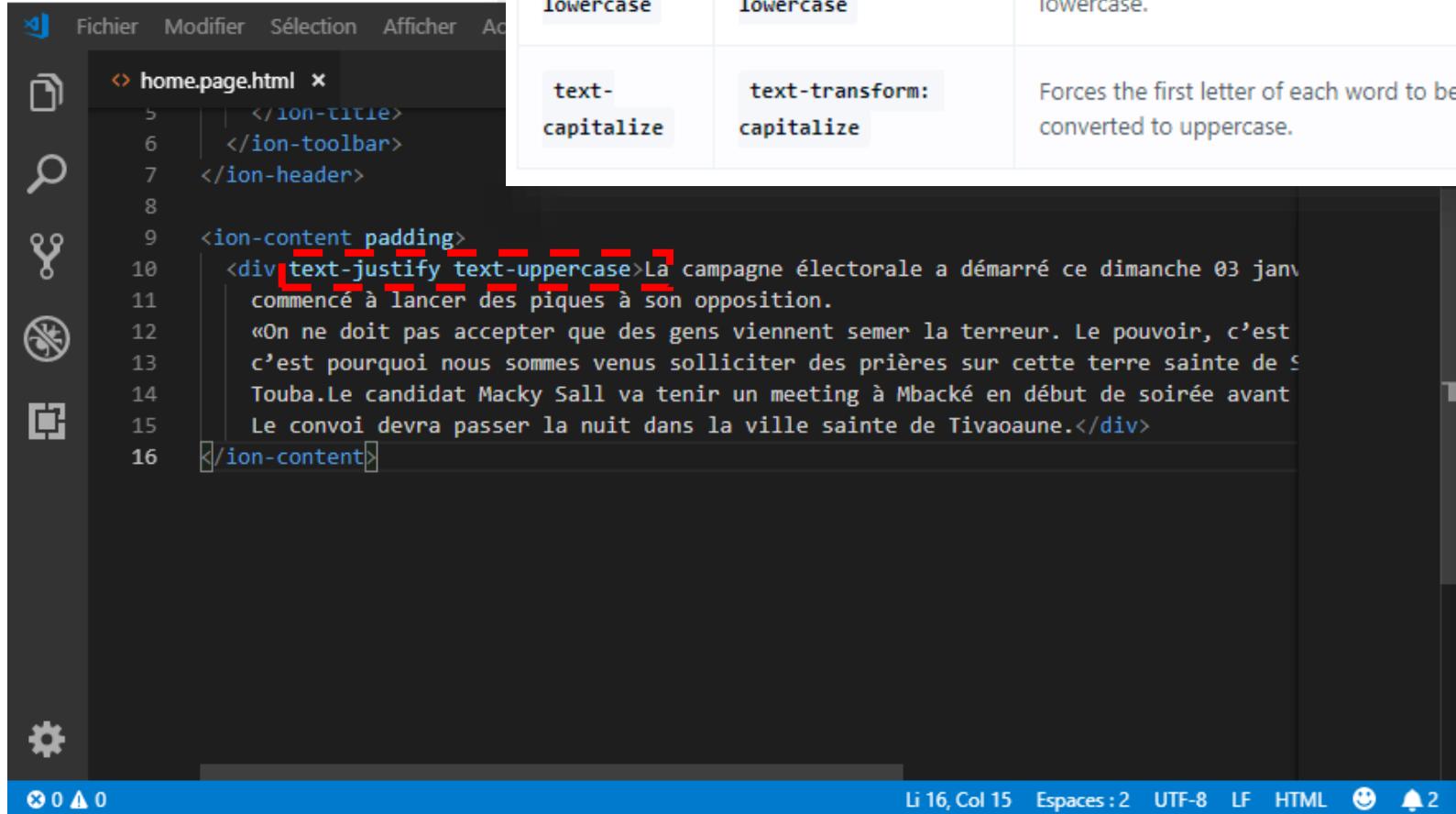
2 Ousmane Sonko

3 Idrissa SECK

4 Issa SALL

La campagne électorale a démarré ce dimanche 03 janvier. Et déjà, le chef de l'Etat, Macky Sall, a commencé à lancer des piques à son opposition. «On ne doit pas accepter que des gens viennent semer la terreur. Le pouvoir, c'est Dieu qui le donne à qui il veut, c'est pourquoi nous sommes venus solliciter des prières sur cette terre sainte de Serigne Touba», a-t-il déclaré à Touba. Le candidat Macky Sall va tenir un meeting à Mbacké en début de soirée avant de faire cap sur Darou Mousty. Le convoi devra passer la nuit dans la ville sainte de Tivaouane.

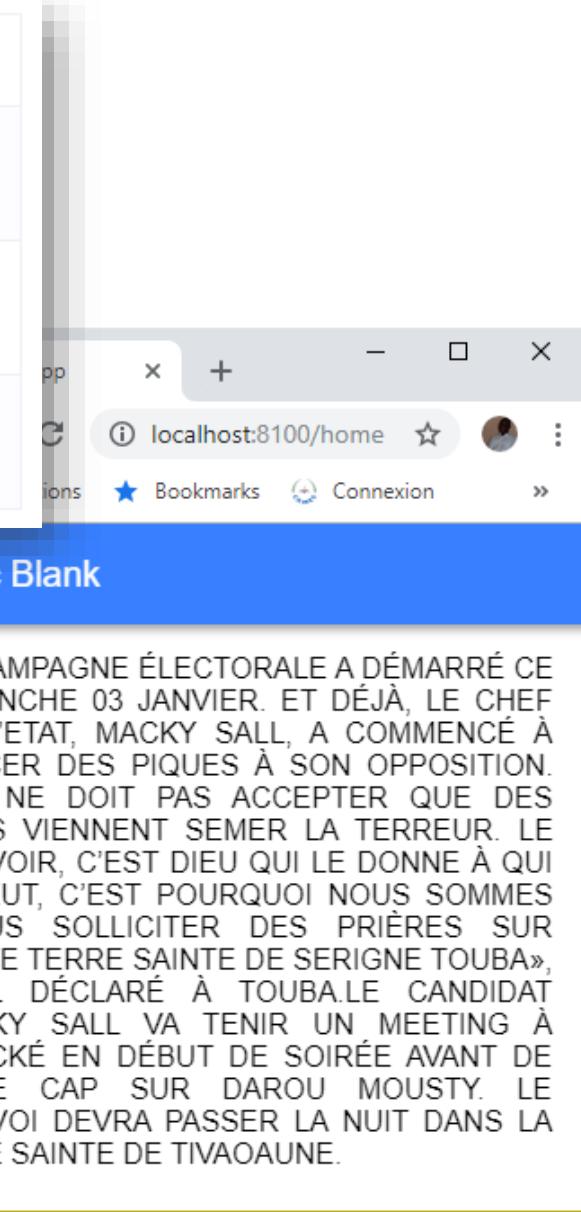
# Personnalisation thème: transformation du texte



The screenshot shows a code editor with the file `home.page.html` open. The code uses the Ionic framework's text transformation classes (`text-uppercase`, `text-lowercase`, `text-capitalize`) to style the text content.

```
<ion-content padding>
  <div text-justify text-uppercase>La campagne électorale a démarré ce dimanche 03 janv
    commencé à lancer des piques à son opposition.
    «On ne doit pas accepter que des gens viennent semer la terreur. Le pouvoir, c'est
    c'est pourquoi nous sommes venus solliciter des prières sur cette terre sainte de S
    Touba.Le candidat Macky Sall va tenir un meeting à Mbacké en début de soirée avant
    Le convoi devra passer la nuit dans la ville sainte de Tivaouane.</div>
</ion-content>
```

Attribute	Style Rule	Description
<code>text-uppercase</code>	<code>text-transform: uppercase</code>	Forces all characters to be converted to uppercase.
<code>text-lowercase</code>	<code>text-transform: lowercase</code>	Forces all characters to be converted to lowercase.
<code>text-capitalize</code>	<code>text-transform: capitalize</code>	Forces the first letter of each word to be converted to uppercase.



# Unités de mesure CSS : em, px, pt, cm, in...

<https://www.w3.org/Style/Examples/007/units.fr.tpl>

- CSS offre différentes unités pour exprimer les dimensions. Certaines proviennent de la typographie, comme le point (pt) et le pica (pc), d'autres sont connues pour leur usage quotidien, comme le centimètre (cm) et le pouce (in). Il y a également une unité "magique" inventée spécifiquement pour CSS: le pixel px
- Pas de restriction à utiliser telle unité à tel ou tel endroit. Si une propriété accepte une valeur en px ('margin: 5px') elle accepte également une valeur en pouces ou en centimètres ('margin: 1.2in; margin: 0.5cm') et vice-versa. La table ci-dessous recommande les différents usages:

	Recommandé	Usage occasionnel	Non recommandé
Écran	em, px, %	ex	pt, cm, mm, in, pc
Imprimante	em, cm, mm, in, pt, pc, %	px, ex	

La relation entre les unités absolues est la suivante: 1in = 2.54cm = 25.4mm = 72pt = 6pc

# Personnalisation thème: **Padding**

- L'attribut padding définit la zone de remplissage d'un élément. La zone de remplissage est l'espace entre le contenu de l'élément et sa bordure.
- La quantité de remplissage à appliquer par défaut est 16px et est définie par la variable --ion-padding.

Attribute	Style Rule	Description
padding	padding: 16px	Applies padding to all sides.
padding-top	padding-top: 16px	Applies padding to the top.
padding-start	padding-start: 16px	Applies padding to the start.
padding-end	padding-end: 16px	Applies padding to the end.
padding-bottom	padding-bottom: 16px	Applies padding to the bottom.
padding-vertical	padding: 16px 0	Applies padding to the top and bottom.
padding-horizontal	padding: 0 16px	Applies padding to the left and right.
no-padding	padding: 0	Applies no padding to all sides.

# Personnalisation thème: Padding

Fichier Modifier Sélection Afficher Accéder Déboguer Terminal ... home.page.html - projet1 - Visual Studio...

home.page.html

```
5   |     </ion-title>
6   |     </ion-toolbar>
7   |   </ion-header>
8
9   <ion-content padding>
10  <div padding text-justify text-uppercase>La campagne électorale a démarré ce dimanche
11    Macky Sall, a
12      commencé à lancer des piques à son opposition.
13      «On ne doit pas accepter que des gens viennent semer la terreur. Le pouvoir, c'est
14      c'est pourquoi nous sommes venus solliciter des prières sur cette terre sainte de S
15      Touba.Le candidat Macky Sall va tenir un meeting à Mbacké en début de soirée avant
16      Le convoi devra passer la nuit dans la ville sainte de Tivaouane.</div>
17  </ion-content>
```

Li 10, Col 15 Espaces : 2 UTF-8 LF HTML ☺ 2

Ionic App localhost:8100/home

Applications Bookmarks Connexion

## Ionic Blank

LA CAMPAGNE ÉLECTORALE A DÉMARRÉ CE DIMANCHE 03 JANVIER. ET DÉJÀ, LE CHEF DE L'ETAT, MACKY SALL, A COMMENCÉ À LANCER DES PIQUES À SON OPPOSITION. «ON NE DOIT PAS ACCEPTER QUE DES GENS VIENNENT SEMER LA TERREUR. LE POUVOIR, C'EST DIEU QUI LE DONNE À QUI IL VEUT, C'EST POURQUOI NOUS SOMMES VENUS SOLICITER DES PRIÈRES SUR CETTE TERRE SAINTE DE SERIGNE TOUBA», A-T-IL DÉCLARÉ À TOUBA.LE CANDIDAT MACKY SALL VA TENIR UN MEETING À MBACKÉ EN DÉBUT DE SOIREE AVANT DE FAIRE CAP SUR DAROU MOUSTY. LE CONVOI DEVRA PASSER LA NUIT DANS LA VILLE SAINTE DE TIVAOAUNE.

# Personnalisation thème: Margin

- La zone de marge étend la zone de bordure avec une zone vide utilisée pour séparer l'élément de ses voisins.
- La quantité de marge à appliquer par défaut est 16px et est définie par la variable --ion-margin.

Attribute	Style Rule	Description
margin	margin: 16px	Applies margin to all sides.
margin-top	margin-top: 16px	Applies margin to the top.
margin-start	margin-start: 16px	Applies margin to the left.
margin-end	margin-end: 16px	Applies margin to the right.
margin-bottom	margin-bottom: 16px	Applies margin to the bottom.
margin-vertical	margin: 16px 0	Applies margin to the top and bottom.
margin-horizontal	margin: 0 16px	Applies margin to the left and right.
no-margin	margin: 0	Applies no margin to all sides.

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles hybrides

## Composants IU sous Ionic



# Les Composants de l'Interface Utilisateur

A component is just a class that serves as a *controller for the user interface*. It consists of three parts - some TypeScript code, an HTML template, and CSS styles.

- Les composants sont des éléments importants dans Angular.
- L'application est formée par un ensemble de composants.
- Chaque composant peut imbriquer d'autres composants définissant ainsi une structure hiérarchique.
- Le composant racine s'appelle Root Component

Les applications ioniques sont constituées de blocs de construction de haut niveau appelés composants. Les composants vous permettent de construire rapidement une interface pour votre application.

- Ionic est livré avec un certain nombre de composants, notamment des modaux, des popups et des cartes.
- Bien que les composants soient principalement HTML et CSS, certains composants incluent également une fonctionnalité JavaScript.

# Composants IONIC

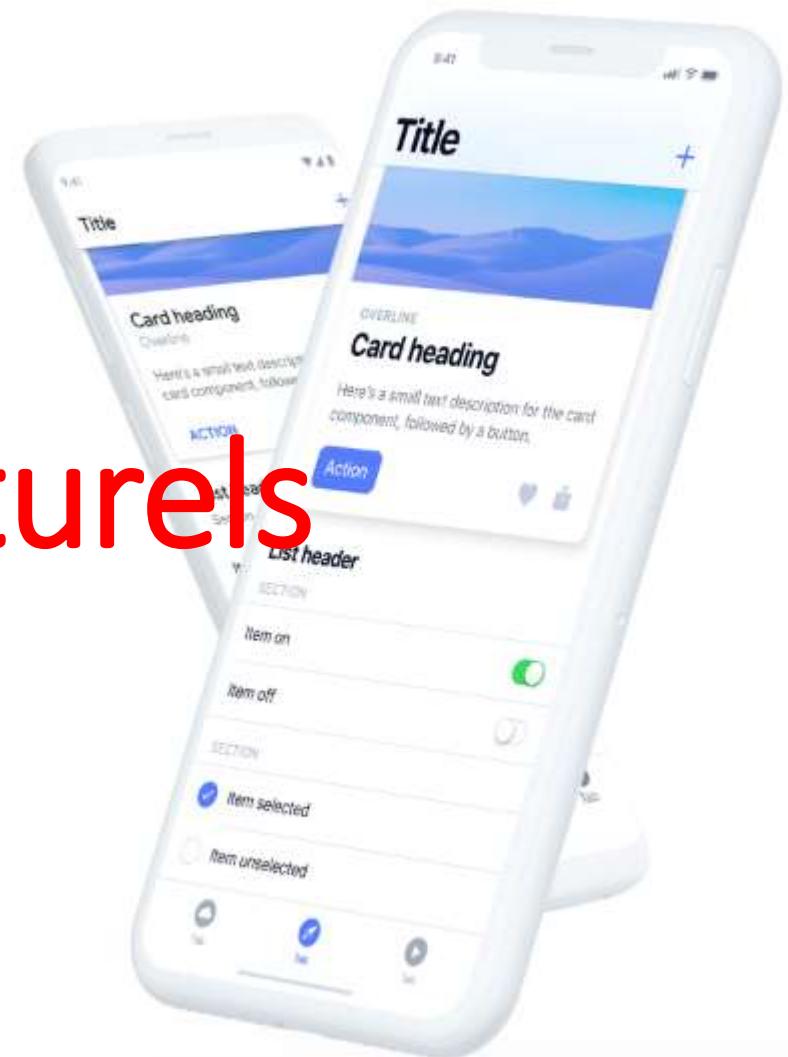
Chaque composant ionic comprend un ou plusieurs éléments personnalisés. Chaque élément personnalisé, à son tour, peut exposer des propriétés, des méthodes, des événements et des propriétés personnalisées CSS.

ion-action-sheet, ion-action-sheet-controller, ion-alert, ion-alert-controller, ion-anchor, ion-app, ion-avatar, ion-back-button, ion-backdrop, ion-badge, ion-button, ion-buttons, ion-card, ion-card-content, ion-card-header, ion-card-subtitle, ion-card-title, ion-checkbox, ion-chip, ion-col, ion-content, ion-datetime, ion-fab, ion-fab-button, ion-fab-list, ion-footer, ion-grid, ion-header, ion-icon, ion-img, ion-infinite-scroll, ion-infinite-scroll-content, ion-input, ion-item, ion-item-divider, ion-item-group, ion-item-option, ion-item-options, ion-item-sliding, ion-label, ion-list, ion-list-header, ion-loading, ion-loading-controller, ion-menu, ion-menu-button, ion-menu-controller, ion-menu-toggle, ion-modal, ion-modal-controller, ion-nav, ion-nav-pop, ion-nav-push, ion-nav-set-root, ion-note, ion-picker, ion-picker-column, ion-picker-controller, ion-popover, ion-popover-controller, ion-progress-bar, ion-radio, ion-radio-group, ion-range, ion-refresher, ion-refresher-content, ion-reorder, ion-reorder-group, ion-ripple-effect, ion-route, ion-route-redirect, ion-router, ion-router-outlet, ion-row, ion-searchbar, ion-segment, ion-segment-button, ion-select, ion-select-option, ion-select-popover, ion-skeleton-text, ion-slide, ion-slides, ion-spinner, ion-split-pane, ion-tab-bar, ion-tab-button, ion-text, ion-textarea, ion-thumbnail, ion-title, ion-toast, ion-toast-controller, ion-toggle, ion-toolbar, ion-virtual-scroll

# Types de composants

- Composants Structurels
- Les Items et éléments cliquables
- Icônes
- Zones et champs de saisie de données
- Listes
- Boîtes et éléments Modaux
- Les Menus

# Composants Structurels



# Les composants structurels sous Ionic

- Trois composants structurels sous Ionic :
  - **<ion-content>**: contient le conteneur principal de la page
  - **<ion-header>**: entête de page contenant entre autres le titre
  - **<ion-footer>**: pied de page
- Les en-têtes et les pieds de page peuvent contenir un élément **<ion-toolbar>** avec du texte et des boutons
- Quelques autres composants peuvent remplacer l'en-tête et le pied de page comme **<ion-tabs>** pour un pied de page avec des onglets vers différentes pages

# Les composants structurels sous Ionic:

se en page sous Ionic

The diagram illustrates the structure of an Ionic page. On the left, a screenshot of an Android emulator shows a green header bar labeled "Entête de page", a yellow main content area labeled "Corps", and a red footer bar labeled "Pied de page". To the right of the screenshot is a code editor window displaying the corresponding HTML and CSS code. Red dashed boxes highlight specific parts of the code, corresponding to the visual components in the screenshot. The code is as follows:

```
<ion-header>
  <ion-toolbar text-center color="success">
    <ion-title>
      Entête de page
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding color="warning" text-center>
  Corps
</ion-content>

<ion-footer>
  <ion-toolbar text-center color="danger">
    Pied de page
  </ion-toolbar>
</ion-footer>
```

Prof. Ousmane SALL, Univ. Thiès, SN

Programmation Applications Mobiles

580

# Les composants structurels sous Ionic: ion-content

- Le composant de contenu fournit une zone de contenu facile à utiliser avec quelques méthodes utiles pour contrôler la zone de défilement. Il ne doit y avoir qu'un seul élément ion-content dans un composant de vue. Propriétés:
  - **color** les options par défaut sont: "primary", "secondary", "tertiary", "success", "warning", "danger", "light", "medium", et "dark"
  - **forceOverscroll** : Si la valeur est *true* et que le contenu ne provoque pas de défilement excessif, l'interaction de défilement provoque un rebond. Si le contenu dépasse les limites de *ionContent*, rien ne changera.
  - **Fullscreen** : Si *true*, le contenu défilera derrière les en-têtes et les pieds de page. Cet effet est facilement visible en définissant la barre d'outils sur transparente.
  - **scrollEvents, scrollX, scrollY**

# Les composants structurels sous Ionic: **ion-content**: CSS Custom Properties

Name	Description
--background	Background of the Content
--color	Color of the Content
--keyboard-offset	Keyboard offset of the Content
--offset-bottom	Offset bottom of the Content
--offset-top	Offset top of the Content
--padding-bottom	Padding bottom of the Content
--padding-end	Padding end of the Content
--padding-start	Padding start of the Content
--padding-top	Padding top of the Content

# Les composants structurels sous Ionic: **ion-toolbar**

- Les barres d'outils sont positionnées au-dessus ou au-dessous du contenu. Lorsqu'une barre d'outils est placée dans un **<ion-header>**, elle apparaîtra en haut du contenu et lorsqu'elle se trouvera dans un **<ion-footer>**, elle apparaîtra en bas.
- Le contenu en plein écran défilera derrière une barre d'outils dans un en-tête ou un pied de page.
- Lorsqu'elles sont placées dans un **<ion-content>**, les barres d'outils défilent avec le contenu.
- **Eléments fils:** **ion-toolbar**, **ion-footer**, **ion-header**, **ion-title**, **ion-buttons**, **ion-back-button**

# Les composants structurels sous Ionic: ion-toolbar

**Buttons**

Buttons placed in a toolbar should be placed inside of the `<ion-buttons>` element. The `<ion-buttons>` element can be positioned inside of the toolbar using a named slot. The below chart has a description of each slot.

Slot	Description
<code>secondary</code>	Positions element to the <code>left</code> of the content in <code>ios</code> mode, and directly to the <code>right</code> in <code>md</code> mode.
<code>primary</code>	Positions element to the <code>right</code> of the content in <code>ios</code> mode, and to the far <code>right</code> in <code>md</code> mode.
<code>start</code>	Positions to the <code>left</code> of the content in LTR, and to the <code>right</code> in RTL.
<code>end</code>	Positions to the <code>right</code> of the content in LTR, and to the <code>left</code> in RTL.

# Les composants structurels sous Ionic: ion-toolbar

**Properties**

color	
Description	The color to use from your application's color palette. Default options are: "primary", "secondary", "tertiary", "success", "warning", "danger", "light", "medium", and "dark". For more information on colors, see <a href="#">theming</a> .
Attribute	color
Type	string   undefined

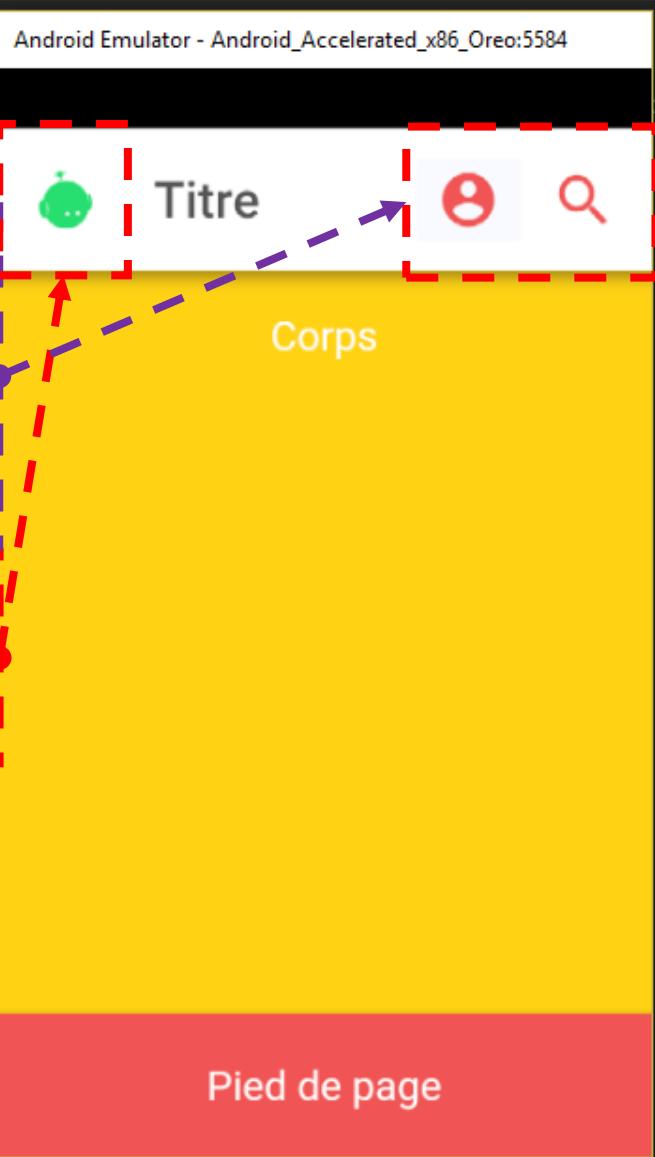
  

mode	
Description	The mode determines which platform styles to use.
Attribute	mode
Type	"ios"   "md"



## home.page.html x

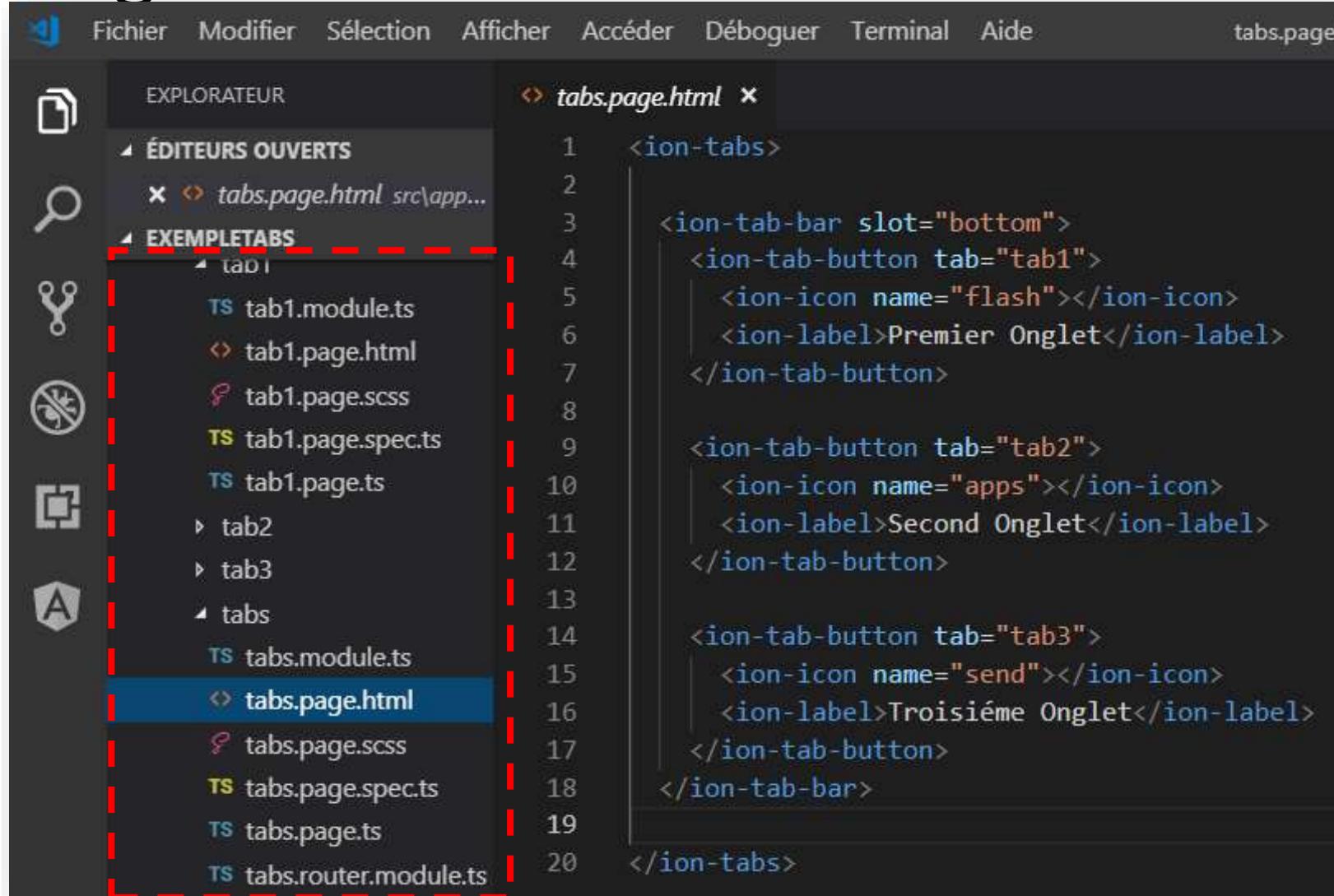
```
1  <ion-header>
2
3    <ion-toolbar>
4      <ion-buttons slot="secondary">
5        <ion-button>
6          | <ion-icon color="danger" slot="icon-only" name="contact"></ion-icon>
7        </ion-button>
8        <ion-button>
9          | <ion-icon slot="icon-only" color="danger" name="search"></ion-icon>
10       </ion-button>
11     </ion-buttons>
12     <ion-buttons slot="start">
13       <ion-button color="secondary">
14         | <ion-icon slot="icon-only" color="success" name="logo-ionitron"></ion-icon>
15       </ion-button>
16     </ion-buttons>
17     <ion-title>Titre</ion-title>
18   </ion-toolbar>
19
20 </ion-header>
21
22 <ion-content padding color="warning" text-center>
23   Corps
24 </ion-content>
25
26 <ion-footer>
27   <ion-toolbar text-center color="danger">
28     | Pied de page
29   </ion-toolbar>
```



# Les composants structurels sous Ionic: les onglets avec **ion-tabs**

- Les onglets sont un composant de navigation de niveau supérieur permettant d'implémenter une navigation à base d'onglets. Le composant est un conteneur de composants individuels **ion-tab**.
- **ion-tabs** est un composant sans style qui fonctionne comme une prise de routeur afin de gérer la navigation. Ce composant ne fournit aucun retour d'interface utilisateur ni aucun mécanisme permettant de basculer entre les onglets. Pour ce faire, une barre d'onglets ion-tab-bar doit être fournie directement comme nœud fils d' **ion-tabs**
- ion-tab
- ion-tab-button

# Les composants structurels sous Ionic: les onglets avec **ion-tabs**

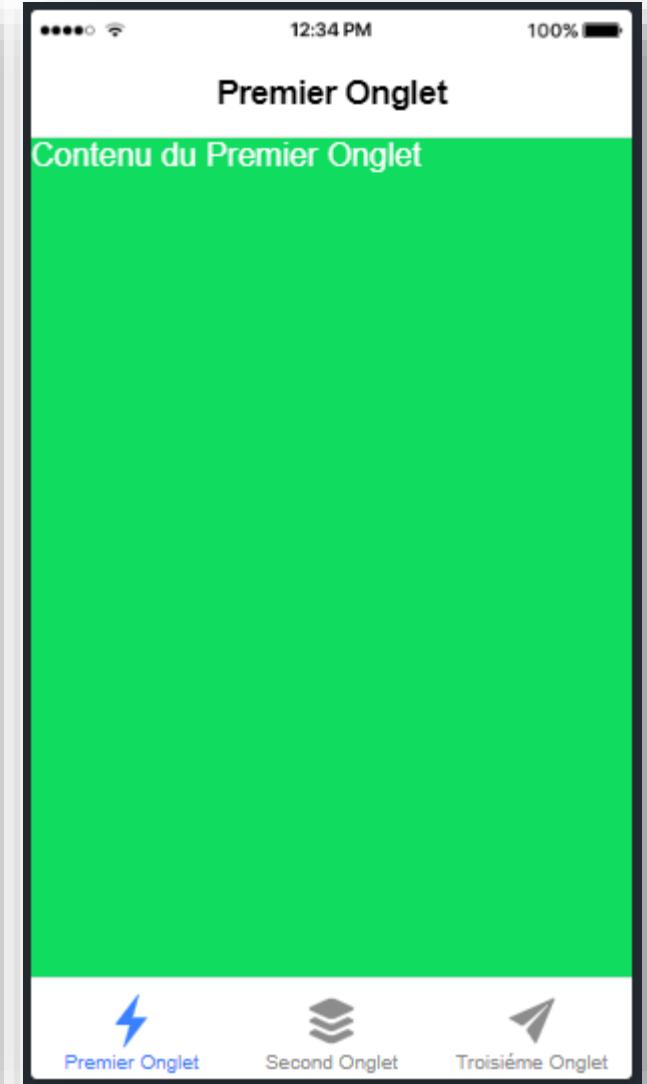


The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists files and folders under 'ÉDITEURS OUVERTS' and 'EXEMPLETAB'. The 'EXEMPLETAB' section is highlighted with a red dashed border. Inside, there are three tabs: 'tab1', 'tab2', and 'tab3'. Under 'tab1', files like 'tab1.module.ts', 'tab1.page.html', 'tab1.page.scss', 'tab1.page.spec.ts', and 'tab1.page.ts' are listed. Under 'tab2', there are 'tab2' and 'tab3'. Under 'tabs', there are 'tabs.module.ts', 'tabs.page.html' (which is currently selected and has a blue background), 'tabs.page.scss', 'tabs.page.spec.ts', 'tabs.page.ts', and 'tabs.router.module.ts'. The main editor area displays the content of 'tabs.page.html'. The code uses the Ionic framework's 'ion-tabs' component to create a tabbed interface. It includes three 'ion-tab-bar' components, each with an 'ion-tab-button' for 'tab1', 'tab2', and 'tab3'. Each button contains an 'ion-icon' and an 'ion-label' with the respective tab name.

```
<ion-tabs>
  <ion-tab-bar slot="bottom">
    <ion-tab-button tab="tab1">
      <ion-icon name="flash"></ion-icon>
      <ion-label>Premier Onglet</ion-label>
    </ion-tab-button>

    <ion-tab-button tab="tab2">
      <ion-icon name="apps"></ion-icon>
      <ion-label>Second Onglet</ion-label>
    </ion-tab-button>

    <ion-tab-button tab="tab3">
      <ion-icon name="send"></ion-icon>
      <ion-label>Troisième Onglet</ion-label>
    </ion-tab-button>
  </ion-tab-bar>
</ion-tabs>
```



# Les composants structurels sous Ionic: **ionic-grid**

<https://www.techiediaries.com/ionic2-grid-system-tutorial/>

<https://ionicframework.com/docs/api/grid>

# Les Items et éléments cliquables



# Les Composants de l'Interface Utilisateur: **ion-item**

- **<ion-item>** est un composant fondamental
- C'est essentiellement un tag HTML **<div>**
  - Peut contenir du texte, des images et d'autres objets
  - A un style css "bloc", donc il apparaît comme une ligne
- Beaucoup d'autres composants doivent être à l'intérieur de **<ion-item>**. Par exemple, **<ion-label>** pour insérer du texte à l'intérieur d'un **<ion-item>**

# Les Composants de l'Interface Utilisateur: **ion-item**

- Les **items** sont des éléments pouvant contenir du texte, des icônes, des avatars, des images, des entrées et tout autre élément natif ou personnalisé. Généralement, ils sont placés dans une liste avec d'autres éléments. Les **items** peuvent être survolés, supprimés, réorganisés, édités, etc.
- Positionnement avec la propriété **slot** :

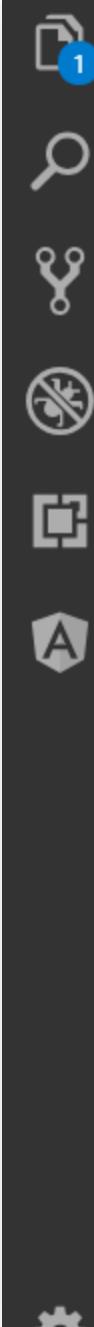
Slot	Description
<code>start</code>	Placed to the left of all other content in LTR, and to the <code>right</code> in RTL.
<code>end</code>	Placed to the right of all other content in LTR, and to the <code>left</code> in RTL.
<code>none</code>	Placed inside of the input wrapper.

# Les Composants de l'Interface Utilisateur: **ion-item**

- Alignement et transformation se fait de la même manière que avec les propriétés du texte pour la personnalisation thème: **Alignement du texte** et **transformation du texte**, vues dans le chapitre précédent.
- Propriétés:
  - **button = "true ou false"** Si la valeur est **true**, une balise de bouton sera rendue et l'élément sera clickable.
  - **color** : avec l'une des neuf couleurs par défaut ou bien un personnalisée.
  - **detail** : Si **true**, une flèche apparaîtra sur l'élément. La valeur par défaut est **false**, sauf si le mode est **ios** et si une propriété **href**, **onclick** ou **button** est présente.
  - **detail-icon** : L'icône à utiliser lorsque **detail** a la valeur **true**.

# Les Composants de l'Interface Utilisateur: **ion-item**

- Propriétés:
  - **disabled= "true ou false "** Si la valeur est **true**
- Eléments d'ion-item
  - **ion-item-divider**(Propriétés: *Color, mode, Sticky*)
  - **ion-item-group** <https://ionicframework.com/docs/api/item-group>
  - **ion-item-sliding** <https://ionicframework.com/docs/api/item-sliding>
  - **ion-item-options** <https://ionicframework.com/docs/api/item-options>
  - **ion-item-option** <https://ionicframework.com/docs/api/item-option>
  - **ion-label** <https://ionicframework.com/docs/api/label>
  - **ion-note** <https://ionicframework.com/docs/api/note>



```
home.page.html
1
2
3
4
5
6
7
8   <ion-content padding>
9
10  <!-- Item par défaut-->
11  <ion-item>
12    | Item
13    | </ion-label>
14  </ion-item>
15  <!-- Item comme Button -->
16  <ion-item (click)="buttonClick()">
17    | Item comme Boutton
18    | </ion-label>
19    | Item comme Boutton
20    | </ion-label>
21  </ion-item>
22  <!-- Item comme lien -->
23  <ion-item href="http://www.seneweb.com">
24    | Aller à Seneweb.com
25    | </ion-label>
26  </ion-item>
27  <ion-item color="secondary">
28    | Item avec la couleur "Secondary"
29    | </ion-label>
30    | Item avec la couleur "Secondary"
31    | </ion-label>
32  </ion-item>
33
34 </ion-content>
```

## Exemple avec les items

Item

Item comme Boutton

Aller à Seneweb.com

Item avec la couleur "Secondary"

64 % 22:27

www.seneweb.com

sene<sup>web</sup>.com Version classique

HOME SOCIÉTÉ SPORTS POLITIQUE PHOTOS



[VIDÉO] RETOUR SUR LA CONF. DE PRESSE DE WADE



DRAME DE TAMBA  
Issa Sall s'explique et accuse...  
Le candidat du Parti de l'Unité du Rassemblement (...)



CAN U 20  
Le Sénégal jouera sa troisième finale d'affilée  
8 minutes

# Les Composants de l'Interface Utilisateur: **ion-button**

- **ion-button** - Applique le style du bouton Ionic au bouton. Propriétés principales:
  - **color:** couleur du bouton "primary", "secondary", "tertiary", "success", "warning", "danger", "light", "medium", and "dark"
  - **disabled:** Si la valeur est true, l'utilisateur ne peut pas interagir avec le bouton.
  - **expand:** Réglez sur "block" pour un bouton de largeur totale ou sur "full" pour un bouton de largeur totale sans bordures gauche et droite.
  - **fill:** Réglez sur "clear" pour un bouton transparent, sur "outline" pour un bouton transparent avec une bordure ou sur "solid". Le style par défaut est "solid" sauf à l'intérieur d'une barre d'outils, où la valeur par défaut est "clear".

# Les Composants de l'Interface Utilisateur: **ion-button**

<https://ionicframework.com/docs/api/button>

- **ion-button** - Applique le style du bouton Ionic au bouton. Propriétés principales:
  - **href**: Contient une URL ou un fragment d'URL vers lequel le lien hypertexte pointe. Si cette propriété est définie, une balise d'ancrage sera rendue.
  - **mode**: pour savoir quelle plateforme utiliser.
  - **routerDirection**: Lors de l'utilisation d'un routeur, il spécifie la direction de la transition lors de la navigation vers une autre page à l'aide de href: "back" | "forward" | "root" .
  - **shape**: La forme du bouton: "round" | undefined .
  - **size**: "default" | "large" | "small" | undefined
  - **strong**: Si true, active un bouton avec un poids de police plus important.
  - **type** : "button" | "reset" | "submit"

# Les Composants de l'Interface Utilisateur: **ion-button**

<https://ionicframework.com/docs/api/button>

<b>Events</b>	
Name	Description
ionBlur	Emitted when the button loses focus.
ionFocus	Emitted when the button has focus.

<b>Slots</b>	
Name	Description
	Content is placed between the named slots if provided without a slot.
"end"	Content is placed to the right of the button text in LTR, and to the left in RTL.
"icon-only"	Should be used on an icon in a button that has no text.
"start"	Content is placed to the left of the button text in LTR, and to the right in RTL.

# Les Composants de l'Interface Utilisateur: ion-button

The screenshot shows a code editor on the left and a browser window on the right. The code editor displays the file `home.page.html` with the following content:

```
<ion-header>
</ion-header>
<ion-content padding>
  <ion-button primary expand="full" shape="round" color="danger">
    <ion-icon name="thumbs-up"></ion-icon>
  </ion-button>
</ion-content>
```

The browser window shows the rendered output: "Ionic Blank" at the top, followed by a large red button with a thumbs-up icon and the text "LABEL DU BOUTON".

The screenshot shows a code editor on the left and a browser window on the right. The code editor displays the file `home.page.html` with the following content:

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Ionic Blank
    </ion-title>
  </ion-toolbar>
</ion-header>
<ion-content padding>
  <ion-button color="danger" href="">
    Label du bouton
    <ion-icon name="thumbs-up" slot="start"></ion-icon>
  </ion-button>
</ion-content>
```

The browser window shows the rendered output: "Ionic Blank" at the top, followed by a red button with a thumbs-up icon and the text "Label du bouton".

# Les Composants de l'Interface Utilisateur: ion-button

The screenshot shows the Visual Studio Code interface with two files open:

- home.page.html**:

```
1 <ion-header>
2   <ion-toolbar color="primary">
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  <ion-button primary expand="full"
11    shape="round" color="danger"
12    (click)="gestionnaireClick($event)">
13    Click n°{{i}}
14  </ion-button>
15 </ion-content>
```
- home.page.ts**:

```
1 import { Component, OnInit } from '@angular/core';
2 import { Key } from 'protractor';
3 import { getComponentViewByIndex } from '@angular/core/src/render3';
4
5 @Component({
6   selector: 'app-home',
7   templateUrl: 'home.page.html',
8   styleUrls: ['home.page.scss'],
9 })
10 export class HomePage {
11   i=0;
12   gestionnaireClick(event: MouseEvent) {
13     this.i++;
14   }
15 }
```

A preview window on the right shows the resulting Ionic application running in a browser. The title bar says "Ionic App". The page has a blue header with the text "Ionic Blank". Below it is a red button with the text "CLICK N°24".

# Les Composants de l'Interface Utilisateur

## button et [ngStyle]



The screenshot shows a mobile application interface. The background is pink, and there is a blue button at the bottom right with the text "CHANGE COLOR". Above the button, the text "Changement de couleur pour #e455aa" is displayed.

```

TS home.page.ts •
  8   templateUrl: 'home.page.html',
  9   styleUrls: ['home.page.scss'],
10 }
11 export class HomePage {
12   public couleur: any = { color: 'danger' };
13   changerCouleur() {
14     this.couleur = randomColor();
15   }
16 }
17
18 function randomColor() {
19   let blue = Math.round(Math.random() * 255);
20   let green = Math.round(Math.random() * 255);
21   let red = Math.round(Math.random() * 255);
22   var rgb = blue | (green << 8) | (red << 16);
23   return '#' + (0x1000000 + rgb).toString(16).slice(1);
24 }
25
26
27

home.page.html •
  1 <ion-header>
  2   <ion-toolbar color="{{couleur}}>
  3     <ion-title>
  4       Ionic Blank
  5     </ion-title>
  6   </ion-toolbar>
  7
  8 </ion-header>
  9
 10 <ion-content padding>
 11   <div expand='full'
 12     [ngStyle]="{{'background-color': couleur}}"
 13     <ion-textarea rows=7 text-center>
 14       Changement de couleur pour {{couleur}}
 15     </ion-textarea>
 16   </div>
 17   <ion-button align-center (click)="changerCouleur()">
 18     change color
 19   </ion-button>
 20 </ion-content>

```

# Les Composants de l'Interface Utilisateur: ion-badge

- Les badges sont des éléments de bloc en ligne qui apparaissent généralement à proximité d'un autre élément. Ils contiennent généralement un nombre ou d'autres caractères.
- Ils peuvent être utilisés pour notifier l'existence d'éléments supplémentaires associés à un élément et indiquer le nombre d'éléments présents.

Properties	
	color
Description	The color to use from your application's color palette. Default options are: "primary" , "secondary" , "tertiary" , "success" , "warning" , "danger" , "light" , "medium" , and "dark" . For more information on colors, see <a href="#">theming</a> .
Attribute	color
Type	string   undefined
mode	
Description	The mode determines which platform styles to use.
Attribute	mode
Type	"ios"   "md"

# Les Composants de l'Interface Utilisateur: **ion-badge**

## CSS Custom Properties

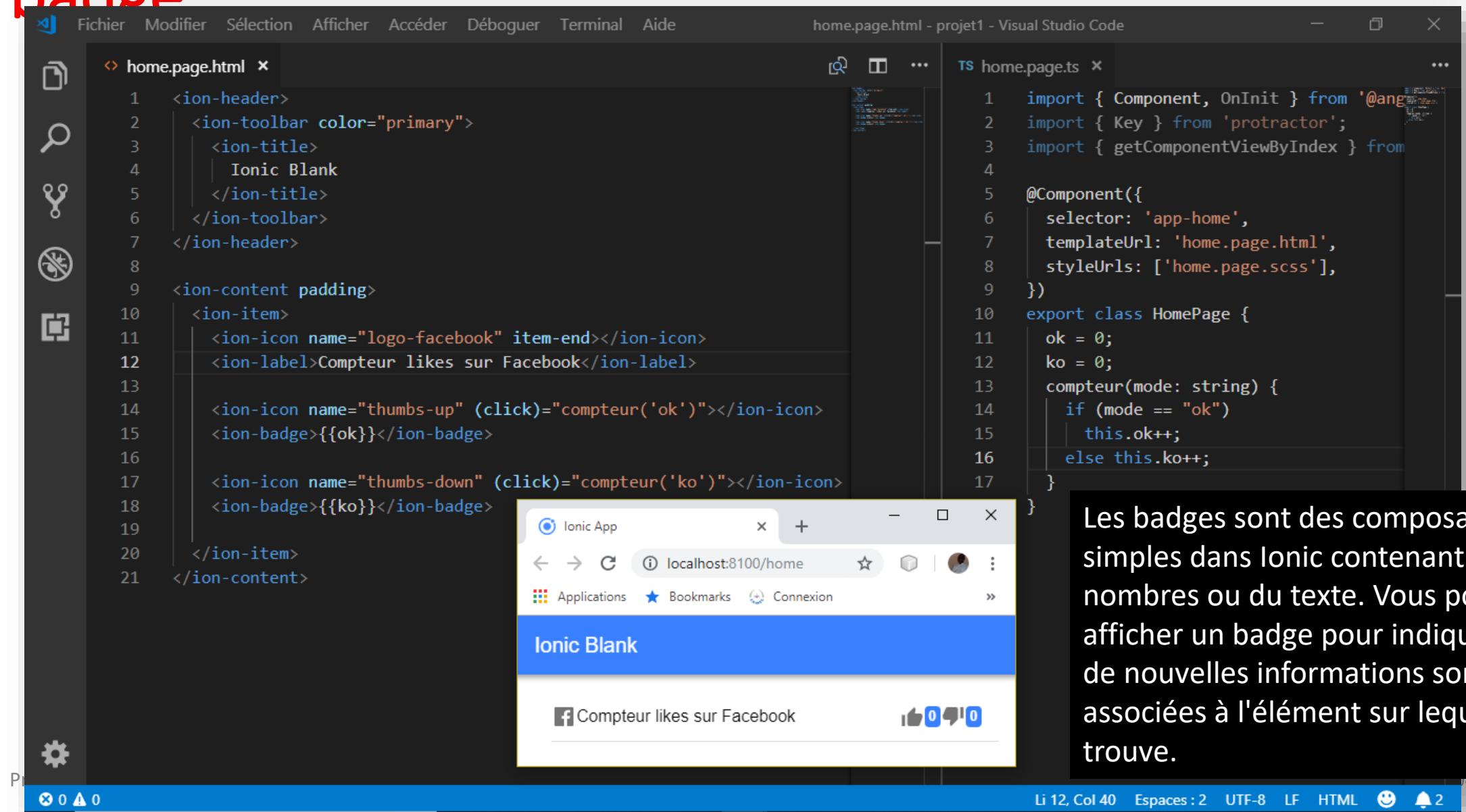
Name	Description
--background	Background of the badge
--color	Text color of the badge
--padding-bottom	Padding bottom of the badge
--padding-end	Padding end of the badge
--padding-start	Padding start of the badge
--padding-top	Padding top of the badge

# Les Composants de l'Interface Utilisateur

## badge

Pour les icônes:

<https://ionicons.com/>



The screenshot shows the Visual Studio Code interface with two files open: `home.page.html` and `home.page.ts`. The `home.page.html` file contains Ionic HTML code with badge components. The `home.page.ts` file contains Angular TypeScript code for a component named `HomePage` that handles a counter. A browser window preview shows the resulting application with a Facebook icon, a "Compteur likes sur Facebook" label, and two badge components showing "0" and "0".

```
home.page.html
<ion-header>
  <ion-toolbar color="primary">
    <ion-title>
      Ionic Blank
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <ion-item>
    <ion-icon name="logo-facebook" item-end></ion-icon>
    <ion-label>Compteur likes sur Facebook</ion-label>

    <ion-icon name="thumbs-up" (click)="compteur('ok')"></ion-icon>
    <ion-badge>{{ok}}</ion-badge>

    <ion-icon name="thumbs-down" (click)="compteur('ko')"></ion-icon>
    <ion-badge>{{ko}}</ion-badge>
  </ion-item>
</ion-content>
```

```
home.page.ts
import { Component, OnInit } from '@angular/core';
import { Key } from 'protractor';
import { getComponentViewByIndex } from 'protractor';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {
  ok = 0;
  ko = 0;
  compteur(mode: string) {
    if (mode == "ok")
      this.ok++;
    else this.ko++;
  }
}
```

Les badges sont des composants simples dans Ionic contenant des nombres ou du texte. Vous pouvez afficher un badge pour indiquer que de nouvelles informations sont associées à l'élément sur lequel il se trouve.

# Les Composants de l'Interface Utilisateur: **ion-card**

- Les cartes sont une des éléments de l'interface utilisateur standard qui sert de point d'entrée pour des informations plus détaillées.
- Une carte peut être un composant unique, mais est souvent composée d'un en-tête, d'un titre, d'un sous-titre et d'un contenu. La carte ionic est divisée en plusieurs sous-composants pour refléter cela:
  - **ion-card-content**: le contenu
  - **ion-card-header**: les entêtes
  - **ion-card-title**: titre
  - **ion-card-subtitle**: sous-titre



```
home.page.html x
10  <ion-card>
11    <ion-card-header>
12      <ion-card-title>Card Title - Président de la République du Sénégal</ion-card-title>
13      <ion-card-subtitle>Card Subtitle - 2019</ion-card-subtitle>
14    </ion-card-header>
15
16    <ion-card-content>
17      
18      Macky Sall est un homme d'État sénégalais, né le 11 décembre 1961 à Fatick,
19      président de la République du Sénégal depuis 2012.
20    </ion-card-content>
21  </ion-card>
22
23  <ion-card>
24    <ion-item>
25      <ion-icon name="pin" slot="start"></ion-icon>
26      <ion-label>ion-item dans une carte, icône à gauche, bouton à droite</ion-label>
27      <ion-button fill="outline" slot="end">View</ion-button>
28    </ion-item>
29
30    <ion-card-content>
31      Ceci est le contenu, sans balises de paragraphe ou en-tête, dans un élément de contenu de carte d'ion
32    </ion-card-content>
33  </ion-card>
34
35  <ion-card>
36    <ion-item href="#" class="activated">
37      <ion-icon name="wifi" slot="start"></ion-icon>
38      <ion-label>Card Link Item 1 .activated</ion-label>
39    </ion-item>
40    <ion-item>
41      <ion-icon name="walk" slot="start"></ion-icon>
42      <ion-label>Card Button Item 2</ion-label>
43    </ion-item>
44  </ion-card>
45 </ion-content>
```

Ionic App × +

localhost:8100/home

Applications Bookmarks Connexion »

## Ionic Blank

Card Title - Président de la République du Sénégal

Card Subtitle - 2019



Macky Sall est un homme d'État sénégalais, né le 11 décembre 1961 à Fatick, président de la République du Sénégal depuis 2012.

ion-item dans une carte, icône à ga... VIEW

Ceci est le contenu, sans balises de paragraphe ou en-tête, dans un élément de contenu de carte d'ion.

Card Link Item 1 .activated

Card Button Item 2



Macky SALL



Ousmane SONKO



Idrissa SECK



Issa SALL BA



Madické NIANG

# Les Composants de l'Interface Utilisateur

## ion-avatar

- Un avatar est un composant qui crée une image circulaire pour un élément.
- Les avatars peuvent être placés à gauche ou à droite d'un élément avec la directive item-left ou item-right.

Code source du fichier home.page.html :

```

Fichier  Modifier  Sélection  Afficher  Accéder  Déboguer  Terminer
home.page.html •
EXPLORATEUR
► ÉDITEURS OUVERTS
▲ PROJET1
  ► e2e
  ► node_modules
  ▲ src
    ► app
      ▲ assets
        ► icon
          □ avatar.png
          □ idrissa.jpg
          □ issa.jpg
          □ macky.jpg
          □ madicke.jpg
          □ shapes.svg
          □ sonko.jpg
1   <ion-header>
2     <ion-toolbar color="primary">
3       <ion-title>
4         Ionic Blank
5       </ion-title>
6     </ion-toolbar>
7   </ion-header>
8
9   <ion-content padding>
10    <ion-list>
11      <ion-item-divider color="primary">
12        Liste des candidats à la présidentielle
13      </ion-item-divider>
14      <ion-item *ngFor="let pers of candidats;">
15        <ion-avatar style="width:75px; height: 75px;">
16          
17        </ion-avatar>&nbsp;
18        <ion-label>{{pers.nom}}</ion-label>
19      </ion-item>
20    </ion-list>
21  </ion-content>
22

```

# Les Composants de l'Ir puces **ion-chips**

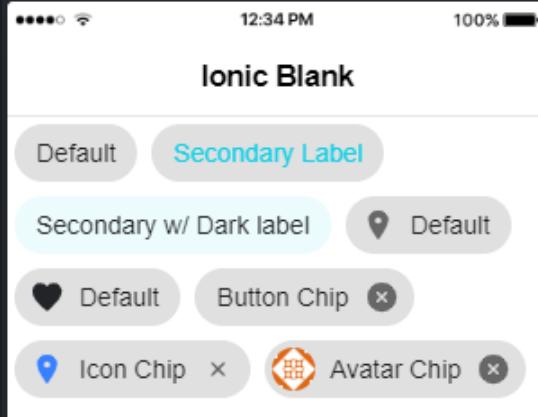
- Les puces représentent des entités complexes dans de petits blocs, tels qu'un contact.
- Une puce peut contenir plusieurs éléments différents tels que des avatars, du texte et des icônes.

## Input Properties

Attr	Type	Details
color	string	The predefined color to use. For example: <code>"primary"</code> , <code>"secondary"</code> , <code>"danger"</code> .
mode	string	The mode to apply to this component. Mode can be <code>ios</code> , <code>wp</code> , or <code>md</code> .

home.page.html

```
9  <ion-content>
10 <ion-chip>
11   | <ion-label>Default</ion-label>
12 </ion-chip>
13
14 <ion-chip>
15   | <ion-label color="secondary">Secondary Label</ion-label>
16 </ion-chip>
17
18 <ion-chip color="secondary">
19   | <ion-label color="dark">Secondary w/ Dark label</ion-label>
20 </ion-chip>
21
22 <ion-chip>
23   | <ion-icon name="pin"></ion-icon>
24   | <ion-label>Default</ion-label>
25 </ion-chip>
26
27 <ion-chip>
28   | <ion-icon name="heart" color="dark"></ion-icon>
29   | <ion-label>Default</ion-label>
30 </ion-chip>
31
32 <ion-chip>
33   | <ion-label>Button Chip</ion-label>
34   | <ion-icon name="close-circle"></ion-icon>
35 </ion-chip>
36
37 <ion-chip>
38   | <ion-icon name="pin" color="primary"></ion-icon>
39   | <ion-label>Icon Chip</ion-label>
40   | <ion-icon name="close"></ion-icon>
41 </ion-chip>
42
43 <ion-chip>
44   | <ion-avatar>
45     |   
46   | </ion-avatar>
47   | <ion-label>Avatar Chip</ion-label>
48   | <ion-icon name="close-circle"></ion-icon>
49 </ion-chip>
50
51 </ion-content>
```



# Les Composants de l'Interface Utilisateur: Segment avec **ion-segment-button** et **ion-segment**

- Les boutons de segment sont des groupes de boutons associés à l'intérieur d'un segment. Ils sont affichés sur une ligne horizontale.
- Un bouton de segment peut être sélectionné par défaut en ajoutant l'attribut **checked** ou en définissant la valeur du segment sur la valeur du bouton de segment. Un seul bouton de segment doit être sélectionné à la fois.

Properties	
<b>checked</b>	
Description	If <code>true</code> , the segment button is selected.
Attribute	<code>checked</code>
Type	<code>boolean</code>
<b>disabled</b>	
Description	If <code>true</code> , the user cannot interact with the segment button.
Attribute	<code>disabled</code>
Type	<code>boolean</code>
<b>layout</b>	
Description	Set the layout of the text and icon in the segment.
Attribute	<code>layout</code>
Type	<code>"icon-bottom"   "icon-end"   "icon-hide"   "icon-start"   "icon-top"   "label-hide"   undefined</code>

# Les Composants de l'Interface Utilisateur: Segment avec **ion-segment-button** et **ion-segment**

mode	
Description	The mode determines which platform styles to use.
Attribute	mode
Type	"ios"   "md"
value	
Description	The value of the segment button.
Attribute	value
Type	string

Events	
Name	Description
ionSelect	Emitted when the segment button is clicked.

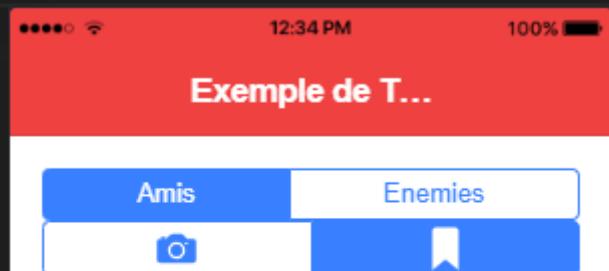
home.page.html x

```
7<ion-content id="contenu" padding>
8
9    <!-- Segment buttons with text and click listeners -->
10   <ion-segment>
11     <ion-segment-button
12       (ionSelect)="segmentButtonClicked($event)">
13       <ion-label>Amis</ion-label>
14     </ion-segment-button>
15     <ion-segment-button
16       (ionSelect)="segmentButtonClicked($event)">
17       <ion-label>Enemies</ion-label>
18     </ion-segment-button>
19   </ion-segment>
20
21   <!-- Segment buttons with values and icons -->
22   <ion-segment>
23     <ion-segment-button value="camera">
24       <ion-icon name="camera"></ion-icon>
25     </ion-segment-button>
26     <ion-segment-button value="bookmark">
27       <ion-icon name="bookmark"></ion-icon>
28     </ion-segment-button>
29   </ion-segment>
30
31 </ion-content>
```

↔ □ ...

ts home.page.ts x

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage {
9   segmentButtonClicked(ev: any) {
10     console.log('Segment button clicked', ev);
11   }
12 }
```



# Les Composants de l'Interface Utilisateur: ion-thumbnail

# Les Composants de l'Interface Utilisateur: Progress Indicators avec ion-loading, ion-loading-controller, ion-progress-bar, ion-skeleton-text, ion-spinner

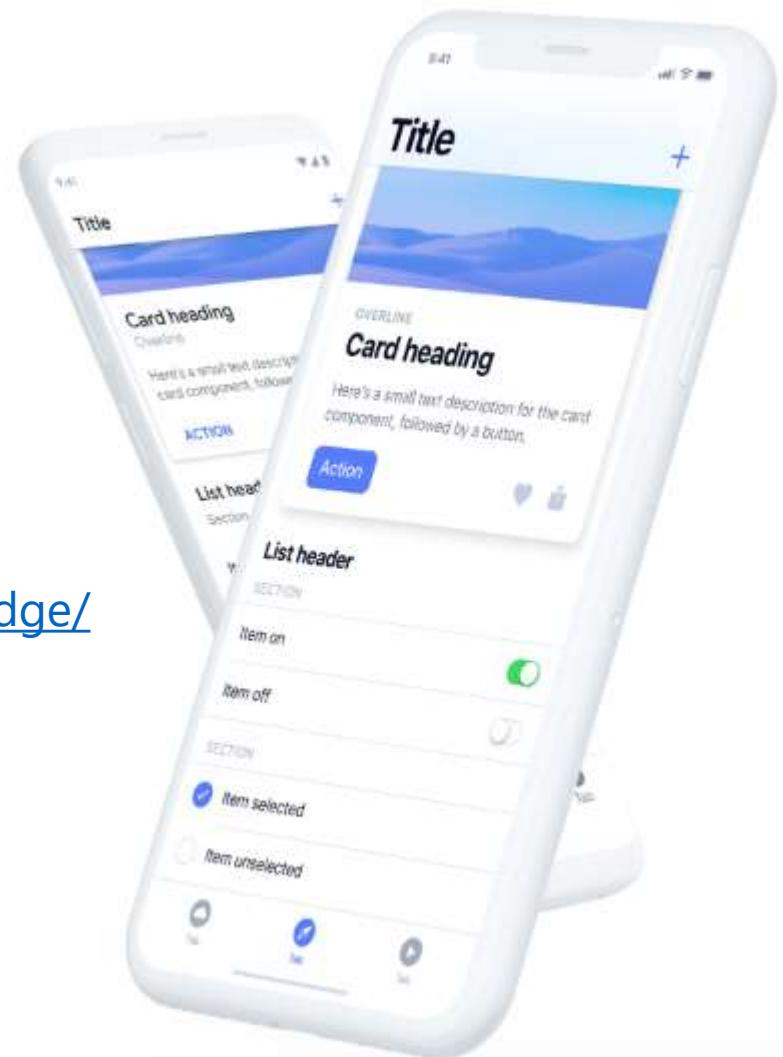
# Les Composants de l'Interface Utilisateur: **Refresher avec ion-refresher-content et ion-refresher**

# Les Composants de l'Interface Utilisateur: Reorder avec **ion-reorder-group** et **ion-reorder**

# Les Composants de l'Interface Utilisateur: Slides avec **ion-slide** et **ion-slides**

# Les Icônes

<https://beta.ionicframework.com/docs/api/badge/>  
<https://ionicons.com/>



```
home.page.html x
1 <ion-header>
2   <ion-toolbar color="danger">
3     | <ion-title>Exemple avec les Icons </ion-title>
4   </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8   <ion-item>
9     | <ion-badge color="primary">2019</ion-badge>
10  </ion-item>
11
12  <ion-item>
13    | <ion-badge color="secondary">Sénégal</ion-badge>
14  </ion-item>
15
16  <ion-item>
17    | <ion-icon name="heart"></ion-icon>
18  </ion-item>
19
20  <ion-item>
21    | <ion-icon name="moon"></ion-icon>
22  </ion-item>
23
24  <ion-item>
25    | <ion-badge color="secondary">
26      |   | <ion-icon name="moon"></ion-icon>
27      |   | </ion-badge>
28    | </ion-item>
29 </ion-content>
```

## Exemple avec les Icons

2019

Sénégal



# Les zones et champs de saisie de données

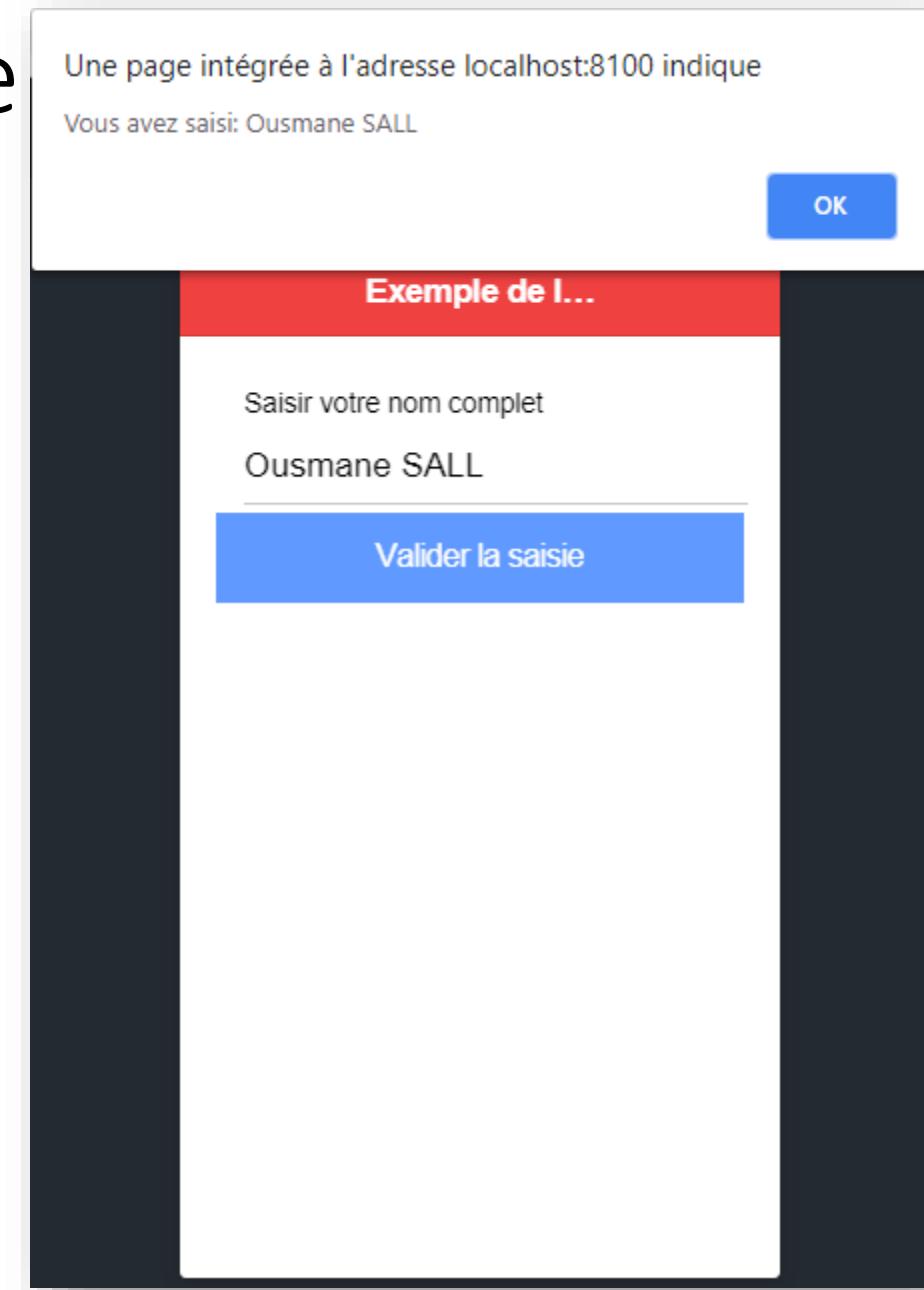


# Les zones de saisie de données

- Ionic fournit beaucoup d'éléments de champs de saisie communs:
  - DateTime
  - Checkbox
  - Text input
  - ...
- Pour la plupart, ils devraient toujours être dans un élément ion-item
- Relié comme dans Angular, avec un binding bidirectionnelle avec **[(ngModel)]**

# Les Composants de l'Interface **ion-input, ion-textarea**

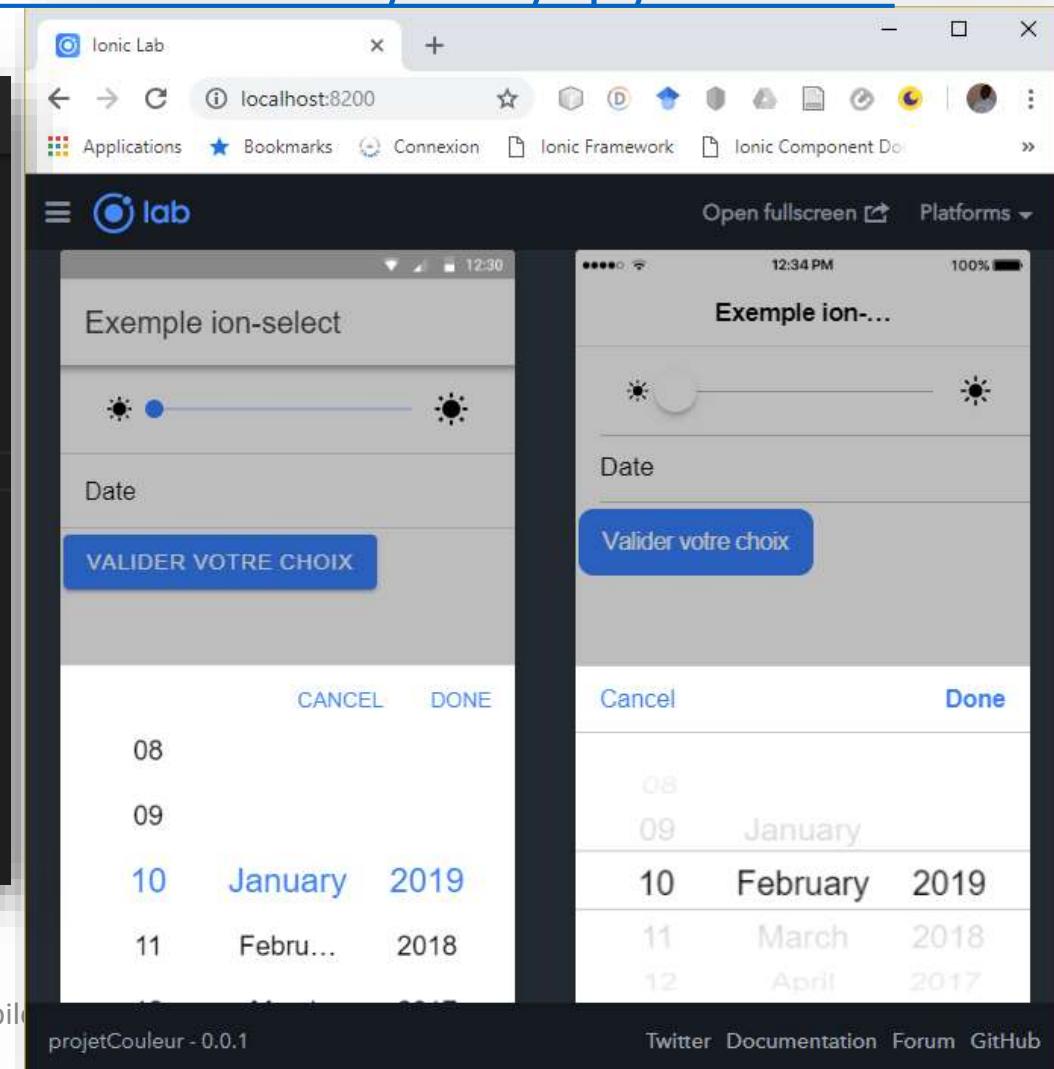
- Le composant input est une enveloppe pour l'élément d'entrée HTML avec un style personnalisé et des fonctionnalités supplémentaires.
- Il accepte la plupart des mêmes propriétés que l'entrée HTML, mais fonctionne très bien sur les appareils de bureau et s'intègre au clavier sur les appareils mobiles.
- Il est destiné aux entrées de type texte uniquement, telles que "text", "password", "email", "number", "search", "tel", and "url".



# Les Composants de l'Interface Utilisateur: Date & Time Pickers avec **ion-datetime**, **ion-picker-controller**, **ion-picker**

<https://ionicframework.com/docs/api/datetime>

```
home.page.html
1  </ion-header>
2
3  <ion-content>
4
5    <ion-item>
6      <ion-label>Date</ion-label>
7      <ion-datetime display-format="DD-MM-YYYY" picker-format="DD MMMM YYYY">
8        </ion-datetime>
9    </ion-item>
10   <ion-button (click)="validerChoix()">
11     Valider votre choix
12   </ion-button><br>
13   <ion-label>
14     {{donnee == null ? '' : 'Vous avez voté pour '+donnee}}
15   </ion-label>
16
17 </ion-content>
```

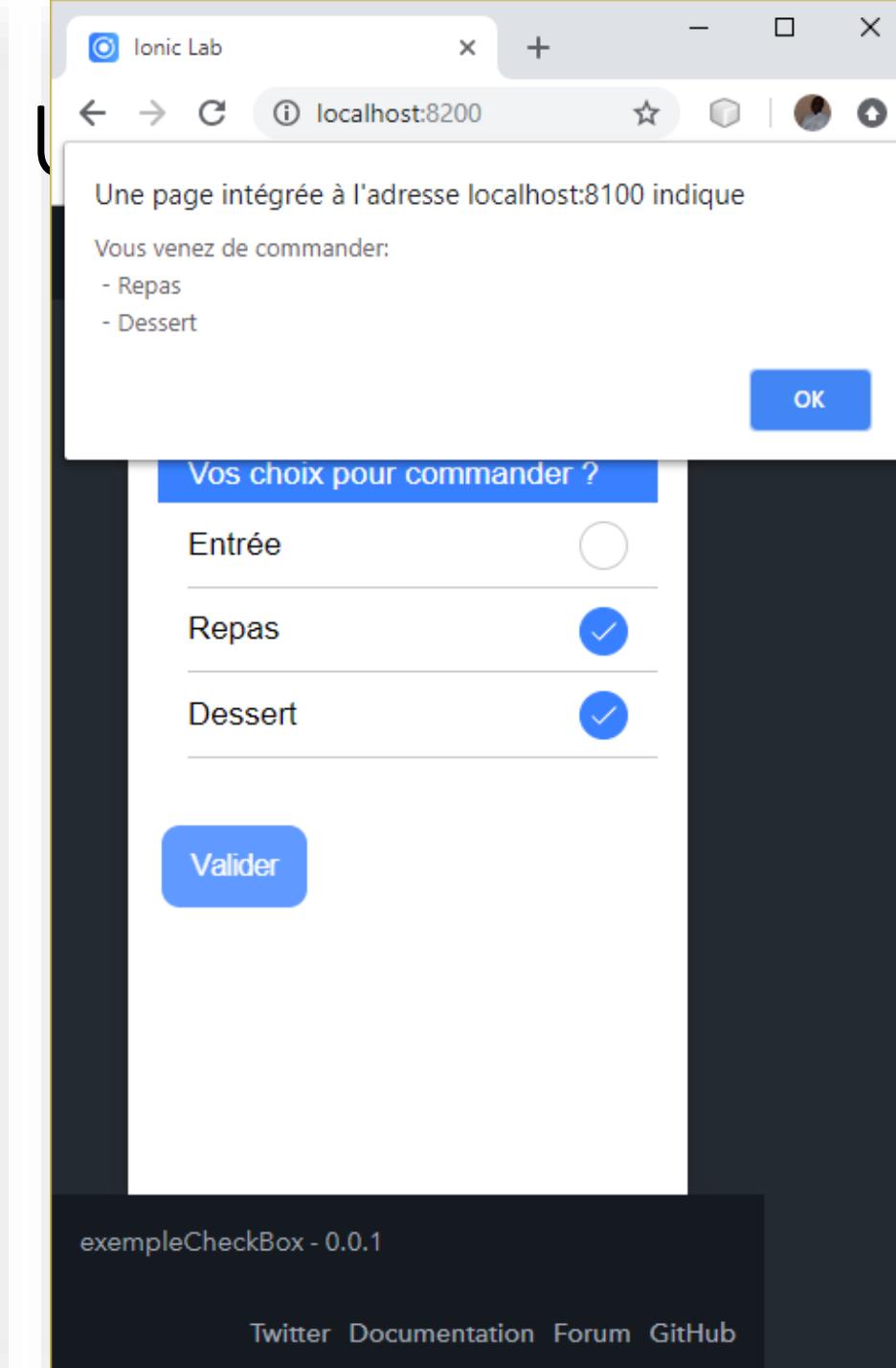
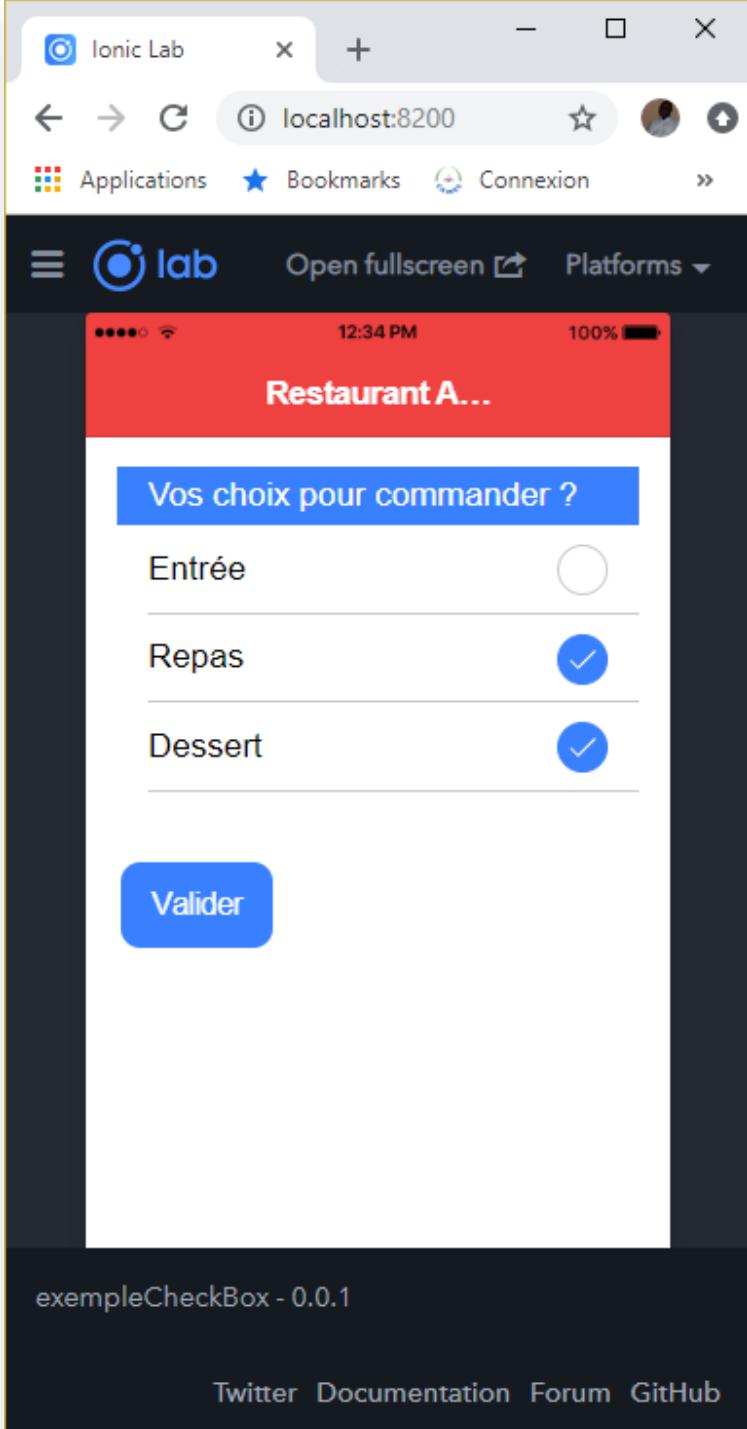


# Les Composants de l'Interface Utilisateur: **ion-checkbox**

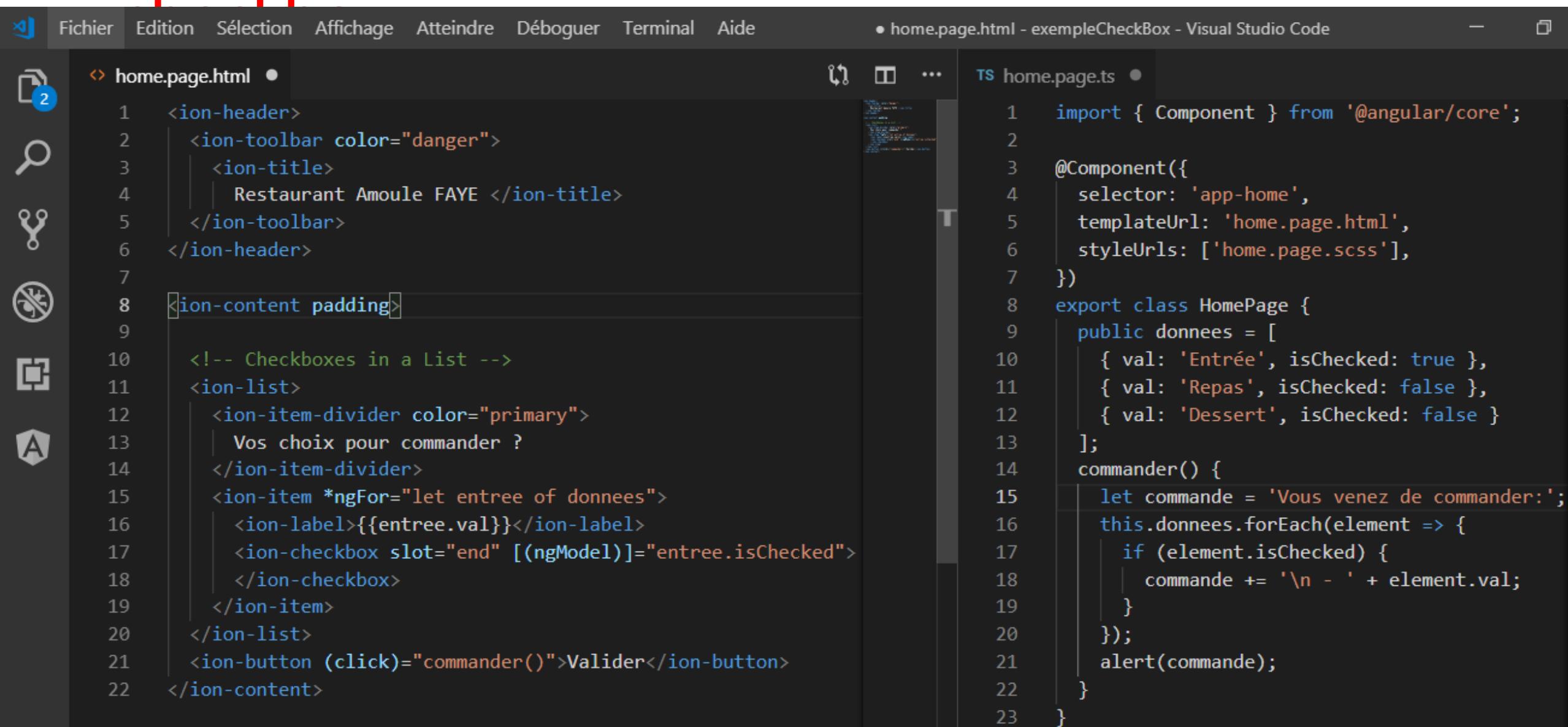
- Les cases à cocher permettent de sélectionner plusieurs options parmi un ensemble d'options. Ils apparaissent cochés lorsqu'ils sont activés.
- En cliquant sur une case à cocher va changer valeur de sa propriété **checked**.
- L'état d'un checkbox peut également être vérifié par programmation en définissant la propriété **checked**.

<https://ionicframework.com/docs/api/checkbox>

# Les Composants checkbox



# Les Composants de l'Interface Utilisateur: ion-



The screenshot shows the Visual Studio Code interface with two files open:

- home.page.html**: An Angular component template file. It includes an ion-header with a title "Restaurant Amoule FAYE", an ion-content section with padding, an ion-list, and an ion-item with an ion-label and an ion-checkbox.
- home.page.ts**: An Angular component class file. It defines a component selector 'app-home', templateUrl 'home.page.html', and styleUrls 'home.page.scss'. The class HomePage has a public variable donnees containing an array of objects representing meal choices. It also has a commander() method that collects checked items into a command string and displays an alert.

```
home.page.html
1 <ion-header>
2   <ion-toolbar color="danger">
3     <ion-title>
4       Restaurant Amoule FAYE </ion-title>
5   </ion-toolbar>
6 </ion-header>
7
8 <ion-content padding>
9
10 <!-- Checkboxes in a List -->
11 <ion-list>
12   <ion-item-divider color="primary">
13     Vos choix pour commander ?
14   </ion-item-divider>
15   <ion-item *ngFor="let entree of donnees">
16     <ion-label>{{entree.val}}</ion-label>
17     <ion-checkbox slot="end" [(ngModel)]="entree.isChecked">
18       </ion-checkbox>
19     </ion-item>
20   </ion-list>
21   <ion-button (click)="commander()">Valider</ion-button>
22 </ion-content>
```

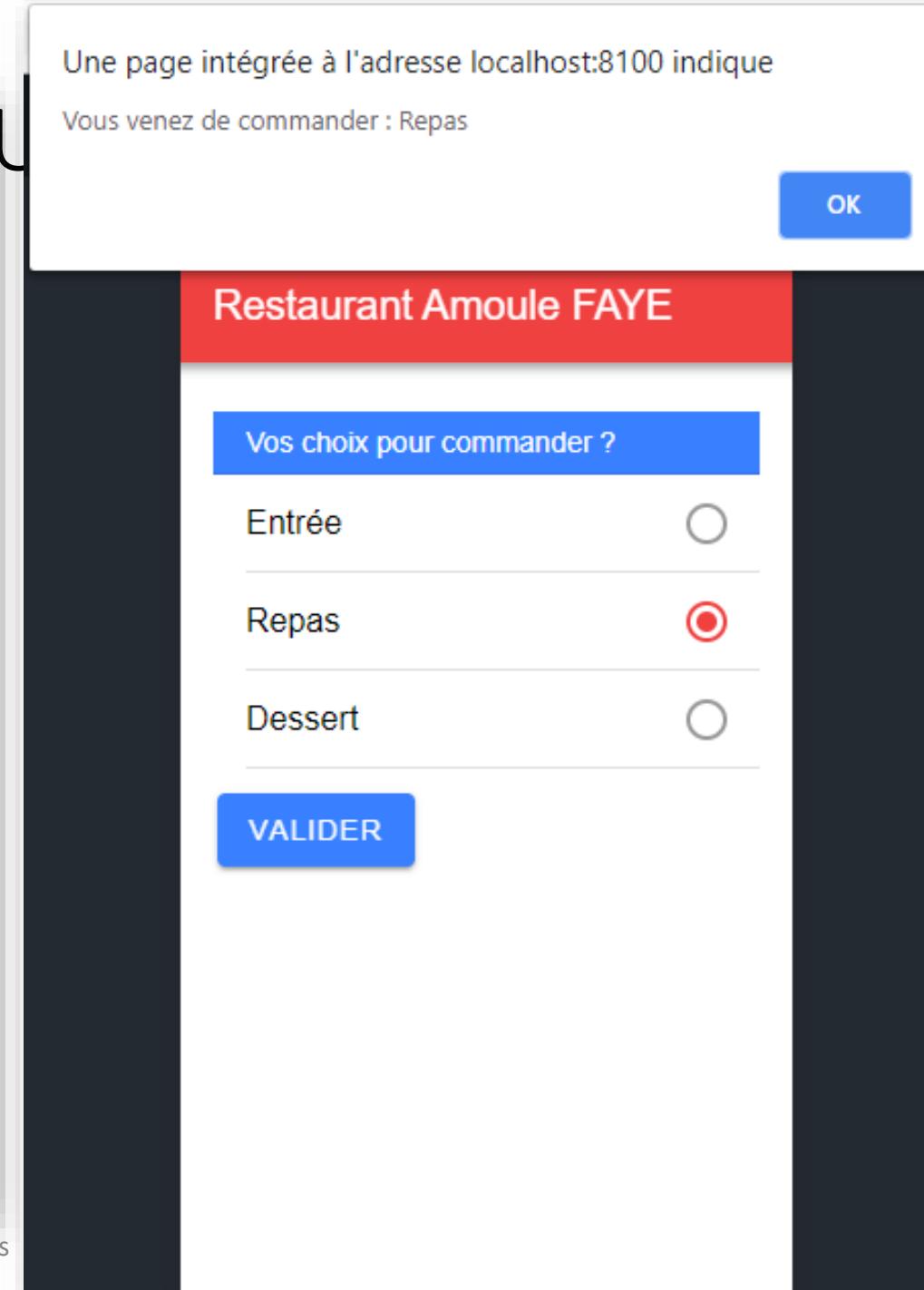
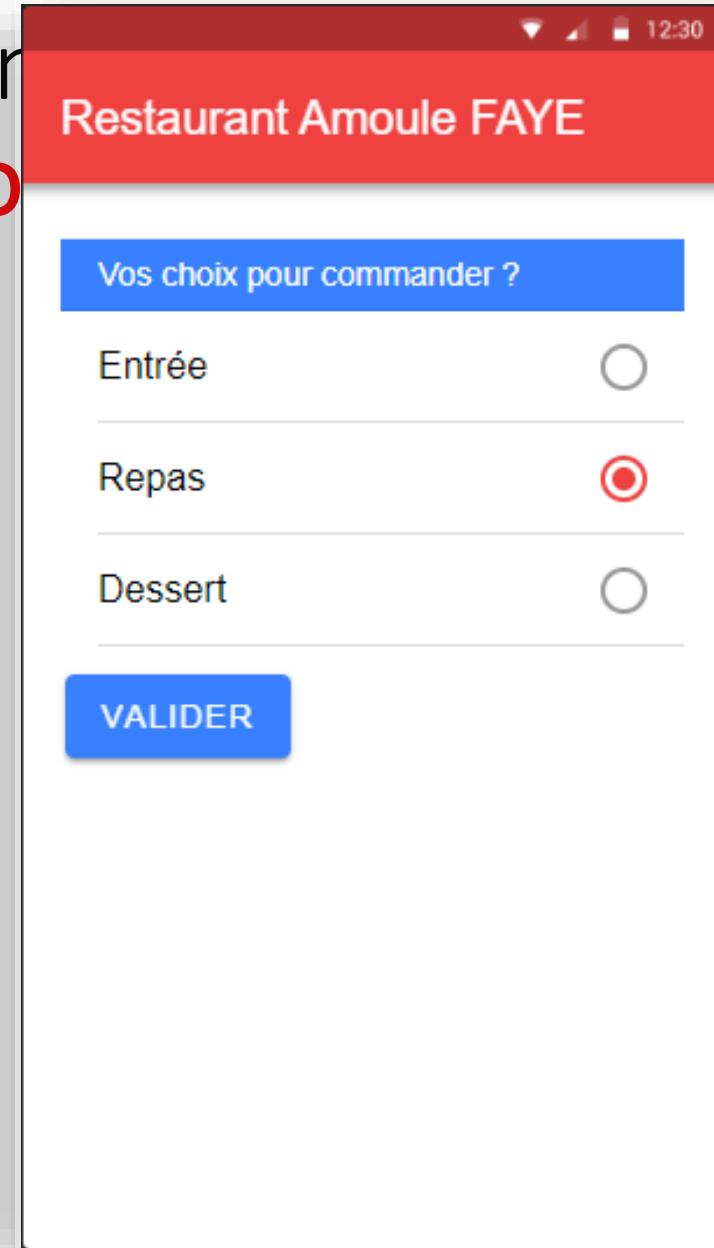
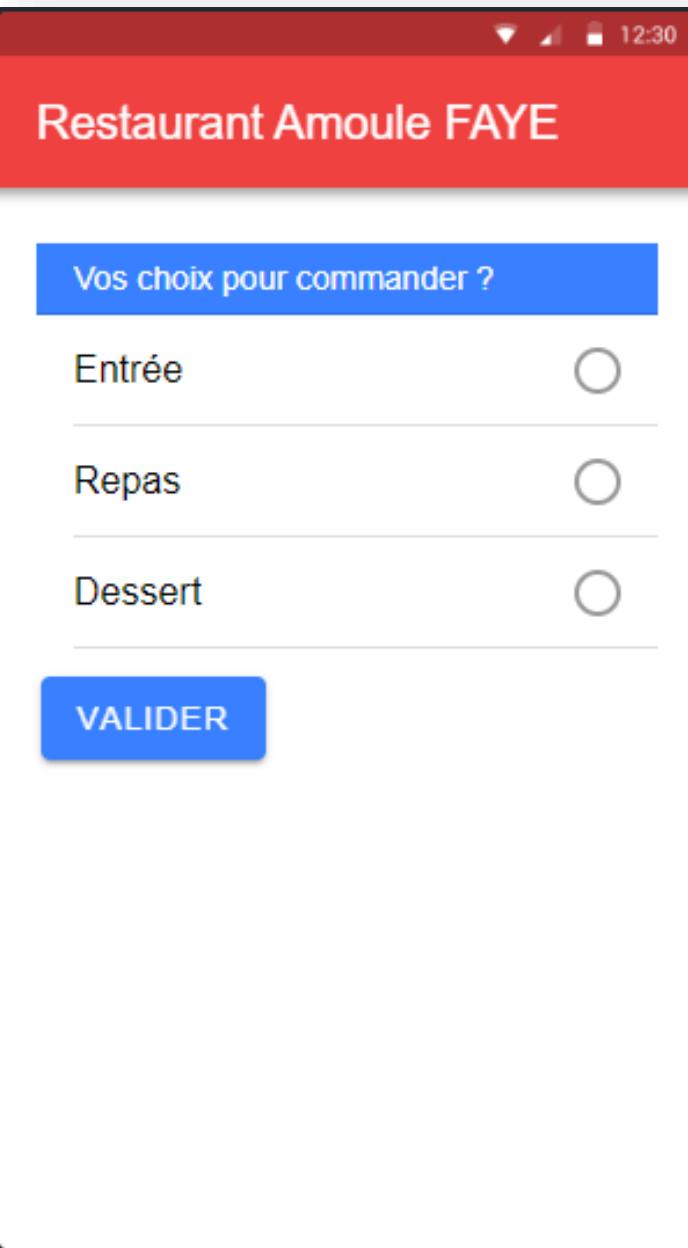
```
home.page.ts
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage {
9   public donnees = [
10     { val: 'Entrée', isChecked: true },
11     { val: 'Repas', isChecked: false },
12     { val: 'Dessert', isChecked: false }
13   ];
14   commander() {
15     let commande = 'Vous venez de commander:';
16     this.donnees.forEach(element => {
17       if (element.isChecked) {
18         commande += '\n - ' + element.val;
19       }
20     });
21     alert(commande);
22   }
23 }
```

# Les Composants de l'Interface Utilisateur: Radio avec ion-radio-group, ion-radio

- Un groupe de radio est un groupe de boutons radio. Il permet à un utilisateur de sélectionner au plus un bouton radio dans un ensemble.
- Le fait de cocher un bouton radio appartenant à un groupe de radio désélectionne tout bouton radio précédemment coché dans le même groupe.

Events	
Name	Description
ionChange	Emitted when the value has changed.

Properties	
<b>allowEmptySelection</b>	
Description	If <code>true</code> , the radios can be deselected.
Attribute	<code>allow-empty-selection</code>
Type	<code>boolean</code>
<b>name</b>	
Description	The name of the control, which is submitted with the form data.
Attribute	<code>name</code>
Type	<code>string</code>
<b>value</b>	
Description	the value of the radio group.
Attribute	<code>value</code>
Type	<code>any</code>





home.page.html x

```
1 <ion-header>
2   <ion-toolbar color="danger">
3     <ion-title>
4       Restaurant Amoule FAYE </ion-title>
5   </ion-toolbar>
6 </ion-header>
7
8 <ion-content padding>
9   <ion-list>
10    <ion-item-divider color="primary">
11      Vos choix pour commander ?
12    </ion-item-divider>
13
14    <ion-radio-group [(ngModel)]="commande">
15      <ion-item *ngFor="let entree of donnees; let i=index">
16        <ion-label>{{entree.val}}</ion-label>
17        <ion-radio value="{{i}}" lot="end" color="danger">
18          </ion-radio>
19        </ion-item>
20    </ion-radio-group>
21
22  </ion-list>
23  <ion-button (click)="commander()">Valider</ion-button>
24 </ion-content>
```

...

home.page.ts x

```
1 import { Component } from '@angular/core';
2 import { IonRadioGroup, IonRadio } from '@ionic/
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   commande = '';
11   public donnees = [
12     { val: 'Entrée' },
13     { val: 'Repas' },
14     { val: 'Dessert' }
15   ];
16   commander() {
17     alert('Vous venez de commander : '
18       + this.donnees[this.commande].val);
19   }
20 }
```

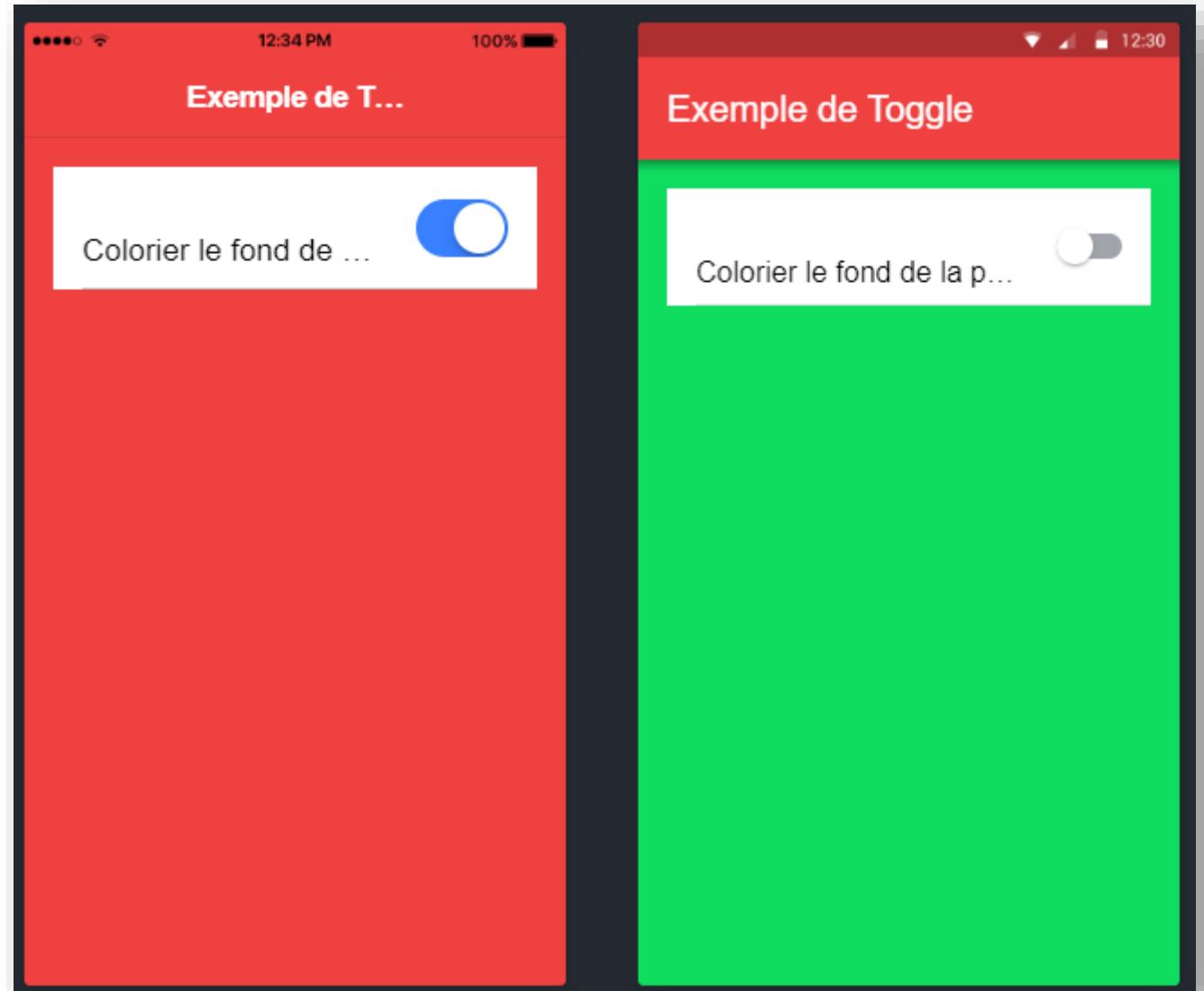


# Les Composants de l'Interface Utilisateur: **Toggle**

- Les éléments **ion-toggle** changent l'état d'une seule option.
- Les **ion-toggle** peuvent être activés ou désactivés en les actionnant ou en les faisant glisser. Ils peuvent également être vérifiés par programme en définissant la propriété **checked**.
- Propriétés:
  - **checked**
  - **color**
  - **disabled**
  - **mode**
  - **name**
  - **value**

# Events	
Name	Description
IonBlur	Emitted when the toggle loses focus.
IonChange	Emitted when the value property has changed.
IonFocus	Emitted when the toggle has focus.

# Les Composants de l'Interface Utilisateur: **Toggle**



# Les Composants de l'Interface Utilisateur: Toggle

The screenshot shows a Visual Studio Code interface with two files open:

- home.page.html**: An Ionic HTML file containing a header with a danger-colored toolbar and a title "Exemple de Toast". Below it is an ion-content section with an ion-item. Inside the ion-item, there is an ion-label with floating position and text "Colorier le fond de la page en rouge". Following the ion-label is an ion-toggle component with an ngModel binding to a bool variable, an ionChange event binding to a changement() function, and a slot="end" attribute.
- home.page.ts**: An Angular TypeScript file defining a HomePage component. The component has a selector 'app-home', templateUrl 'home.page.html', and styleUrls 'home.page.scss'. It contains a bool variable initialized to false and a couleur variable set to 'white'. The changement() method logs the value of bool to the console and then sets the color attribute of the element with id 'contenu' to either 'danger' or 'success' based on the value of bool.

# Les Composants de l'Interface Utilisateur: ion-range

<https://ionicframework.com/docs/api/range>

- Le curseur ion-range permet aux utilisateurs de choisir parmi une plage de valeurs en déplaçant le bouton du curseur. Il peut accepter des boutons doubles, mais par défaut un bouton contrôle la valeur de la plage.
- Propriétés:

- color
- debounce**
- disabled**
- dualKnobs**
- min, max, step
- mode
- mame
- pin
- value

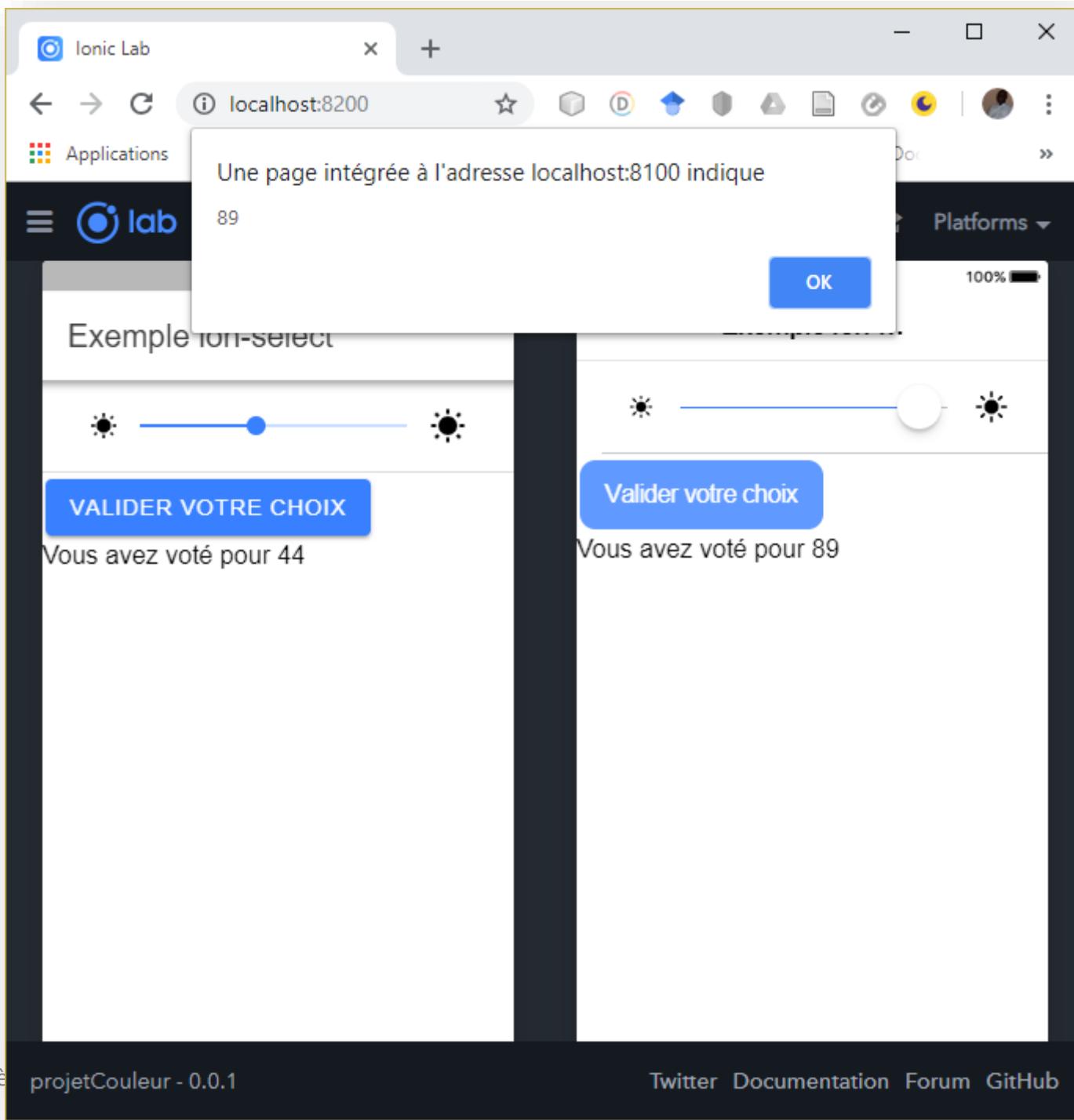
## Events

Name	Description
ionBlur	Emitted when the range loses focus.
ionChange	Emitted when the value property has changed.
ionFocus	Emitted when the range has focus.

## CSS Custom Properties

Name	Description
--bar-background	Background of the range bar
--bar-background-active	Background of the active range bar
--bar-border-radius	Border radius of the range bar
--bar-height	Height of the range bar
--height	Height of the range
--knob-background	Background of the range knob
--knob-border-radius	Border radius of the range knob
--knob-box-shadow	Box shadow of the range knob
--knob-size	Size of the range knob

# Les Com range



eur: ion-



home.page.html x

```
9  <ion-content>
10 <ion-item>
11   <ion-range [(ngModel)]="donnee" value="donnee"
12     (ionChange)="onChange($event)">
13       <ion-icon slot="start" size="small" name="sunny">
14         </ion-icon>
15       <ion-icon slot="end" name="sunny"></ion-icon>
16     </ion-range>
17   </ion-item>
18
19   <ion-button (click)="validerChoix()">
20     Valider votre choix
21   </ion-button><br>
22   <ion-label>
23     {{donnee == null ? '' : 'Vous avez voté pour '+donnee}}
24   </ion-label>
25
26 </ion-content>
```

...  
...

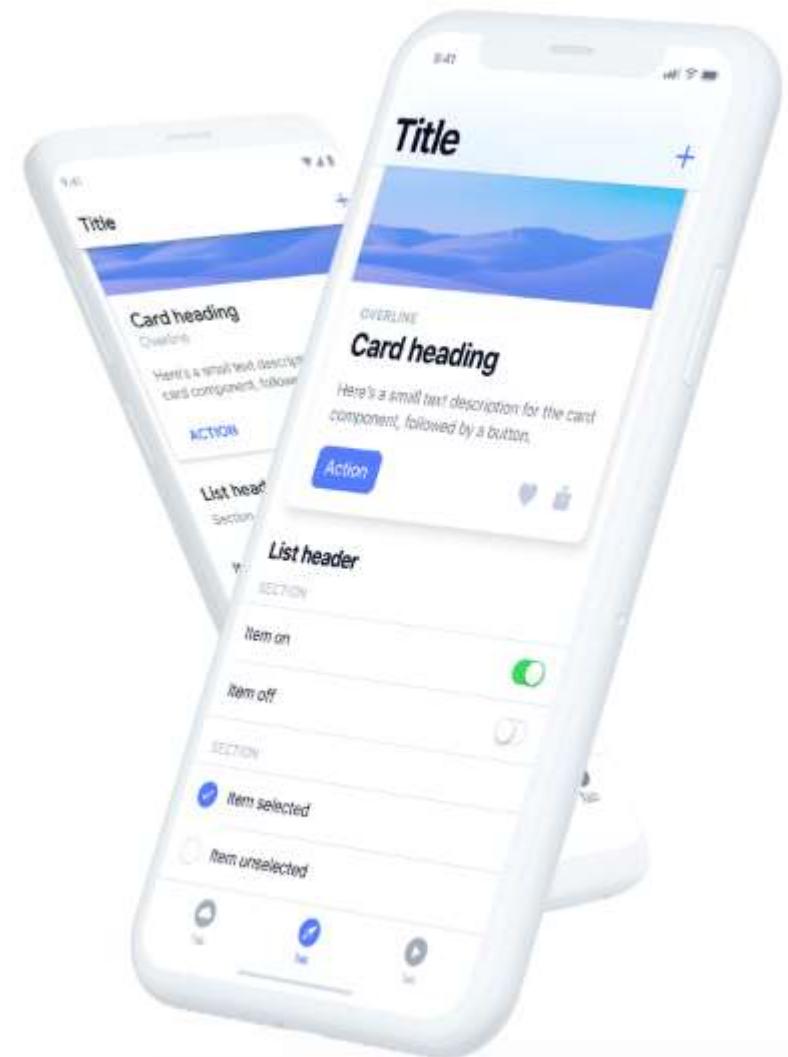
home.page.ts x

```
1  import { Component } from '@angular/core';
2  import { ModalController, IonSelectOption } from '@ionic/angular';
3  import { PageModalePage } from '../page-modale/page-modale.page';
4  import { OverlayEventDetail } from '@ionic/core';
5
6  @Component({
7    selector: 'app-home',
8    templateUrl: 'home.page.html',
9    styleUrls: ['home.page.scss'],
10 })
11 export class HomePage {
12   donnee = null;
13   constructor() {
14   }
15
16   async onChange($event: any) {
17     this.donnee = $event.target.value;
18   }
19
20   validerChoix() {
21     if (this.donnee != null) {
22       alert(this.donnee);
23     }
24   }
25
26 }
27
28
```



# Les Composants de l'Interface Utilisateur: Searchbar avec **ion-searchbar**

# Les Listes



# Les Composants de l'Interface Utilisateur: ion-list

The image shows a code editor with two files and a browser preview.

**home.page.ts**

```
ts home.page.ts x
1 import { Component, OnInit } from '@angular/core';
2 import { Key } from 'protractor';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   noms: string[] = ["Macky SALL",
11   "Ousmane Sonko",
12   "Idrissa SECK",
13   "Issa SALL"];
14 }
15
```

**home.page.html**

```
home.page.html x
1 <ion-toolbar>
2   <ion-title>
3     | Ionic Blank
4   </ion-title>
5 </ion-toolbar>
6 </ion-header>
7
8 <ion-content padding>
9   <ion-list-header color="primary">
10    <ion-label>Liste des candidats à la présidentielle 2019</ion-label>
11  </ion-list-header>
12  <ion-list>
13    <ion-item *ngFor="let nom of noms">{{nom}}</ion-item>
14  </ion-list>
15 </ion-content>
16
```

**Ionic Blank**

Liste des candidats à la présidentielle 2019

- Macky SALL
- Ousmane Sonko
- Idrissa SECK
- Issa SALL



Macky SALL



Ousmane SONKO



Idrissa SECK



Issa SALL BA



Madické NIANG

# Les Composants de l'Interface Utilisateur: ion

Fichier Modifier Sélection Afficher Accéder Déboguer Terminal Aide

EXPLORATEUR

ÉDITEURS OUVERTS

PROJET1

- e2e
- node\_modules
- src
  - app
    - assets
      - icon
        - avatar.png
        - idrissa.jpg
        - issa.jpg
        - macky.jpg
        - madicke.jpg
        - shapes.svg
        - sonko.jpg

home.page.html

```

<ion-header>
  <ion-toolbar color="primary">
    <ion-title>
      Ionic Blank
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <ion-list>
    <ion-item-divider color="primary">
      Liste des candidats à la présidentielle
    </ion-item-divider>
    <ion-item *ngFor="let pers of candidats;">
      <ion-avatar style="width:75px; height: 75px;">
        
      </ion-avatar>&nbsp;
      <ion-label>{{pers.nom}}</ion-label>
    </ion-item>
  </ion-list>
</ion-content>

```

home.page.ts

```

import { Component, OnInit } from '@angular/core';
import { Key, Button } from 'protractor';
import { getComponentViewByIndex } from '@angular/testability';
import { Color } from '@ionic/core';
import { IonItem } from '@ionic/angular';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {
  candidats: Personne[] = [
    new Personne('Macky SALL', 'macky.jpg'),
    new Personne('Ousmane SONKO', 'sonko.jpg'),
    new Personne('Idrissa SECK', 'idrissa.jpg'),
    new Personne('Issa SALL BA', 'issa.jpg'),
    new Personne('Madické NIANG', 'madicke.jpg'),
  ]

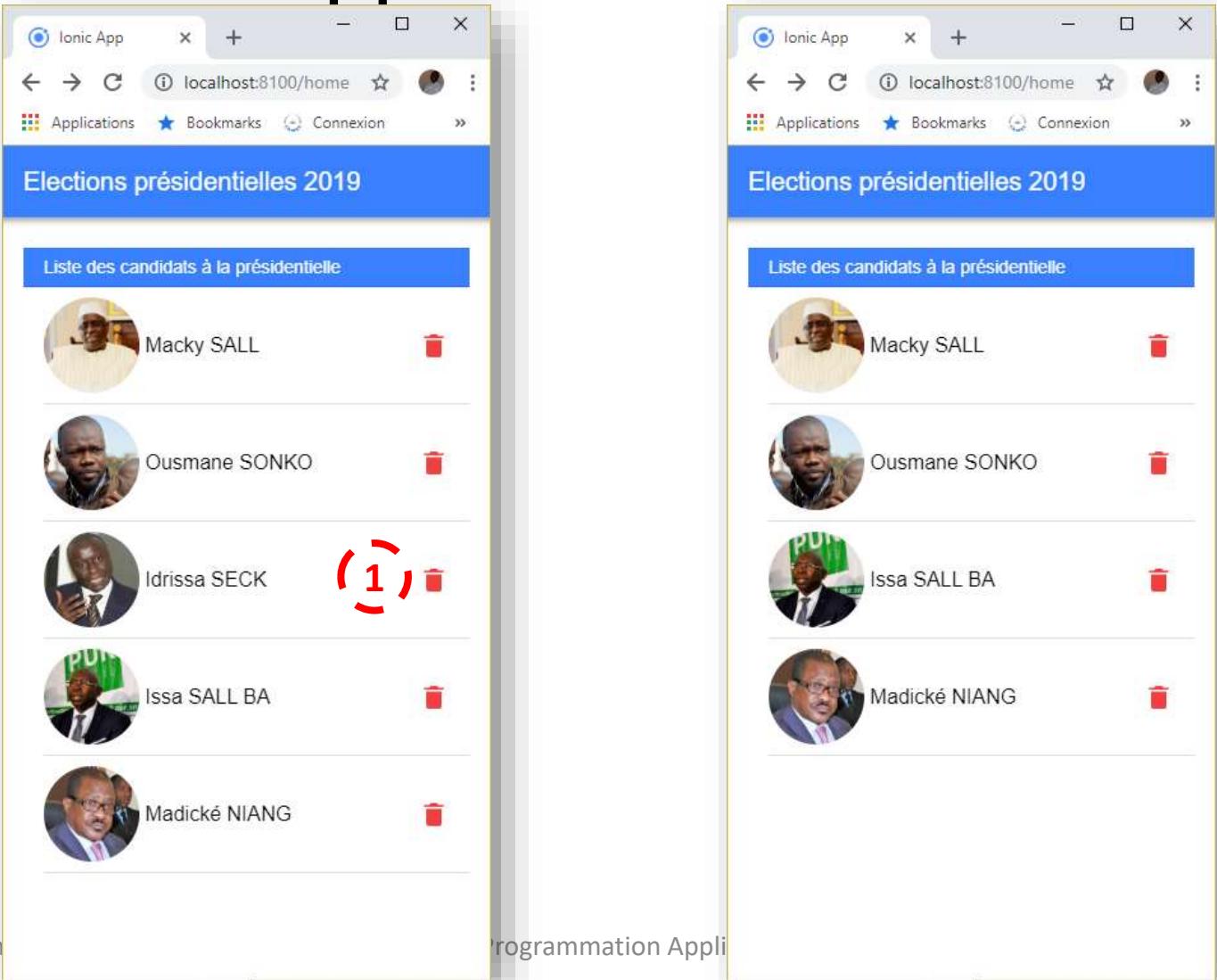
  export class Personne {
    nom: string;
    img: string;
    constructor(nom: string, img: string) {
      this.nom = nom;
      this.img = img;
    }
  }
}

Li 20, Col 1   Espaces : 2   UTF-8   LF   TypeScript 3.2.2   TSLint 1

```

# Les Composants de l'Interface Utilisateur: **ion-list**

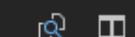
## Gestion de la suppression d'un item





## home.page.html

```
1 <ion-header>
2   <ion-toolbar color="primary">
3     <ion-title>
4       Elections présidentielles 2019
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  <ion-list>
11    <ion-item-divider color="primary">
12      Liste des candidats à la présidentielle
13    </ion-item-divider>
14    <ion-item *ngFor="let pers of candidats; let i=index;">
15
16      <ion-avatar style="width:75px; height: 75px;">
17        
18      </ion-avatar>&nbsp;
19      <ion-label>{{pers.nom}}</ion-label>
20
21      <ion-icon name="trash" color="danger" slot="end" (click)="supprimer(i)">
22        </ion-icon>
23
24    </ion-item>
25  </ion-list>
26 </ion-content>
```



## home.page.ts

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss'],
7 })
8 export class HomePage {
9   candidats: Personne[] = [
10     new Personne('Macky SALL', 'macky.jpg'),
11     new Personne('Ousmane SONKO', 'sonko.jpg'),
12     new Personne('Idrissa SECK', 'idrissa.jpg'),
13     new Personne('Issa SALL BA', 'issa.jpg'),
14     new Personne('Madické NIANG', 'madicke.jpg'),
15
16     ----->
17     supprimer(index: number): void {
18       this.candidats.splice(index, 1);
19     }
20   ]
21
22   export class Personne {
23     nom: string;
24     img: string;
25     constructor(nom: string, img: string) {
26       this.nom = nom;
27       this.img = img;
28     }
29   }
30 }
```

# Les Composants de l'Interface Utilisateur: **ion-list**

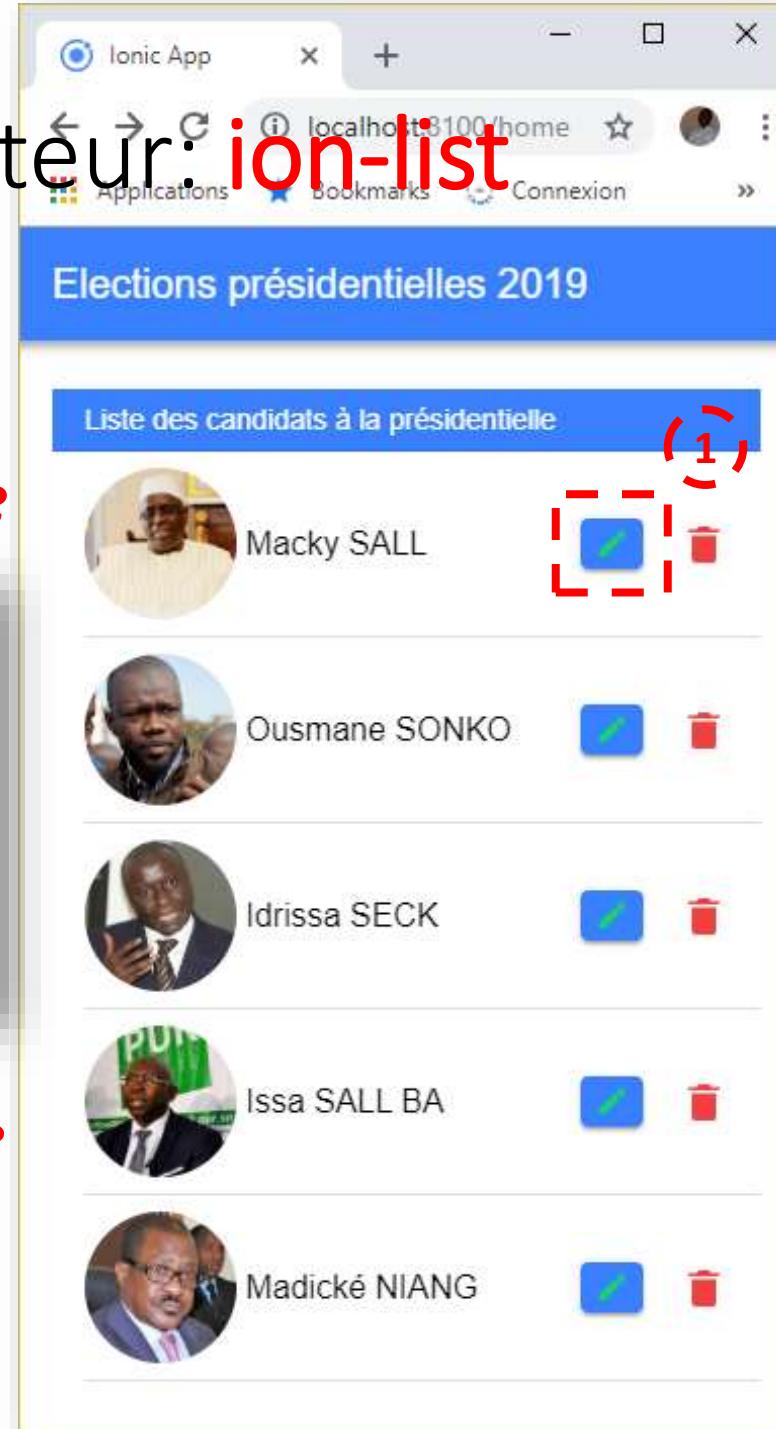
## Gestion de la modification et Camera

Créer un nouveau composant page nommé **secondepage**

```
C:\Users\OSall\Desktop\AppIonic\exempleParams>ionic g page pages/secondepage
> ng generate page pages/secondepage
CREATE src/app/pages/secondepage/secondepage.module.ts (568 bytes)
CREATE src/app/pages/secondepage/secondepage.page.html (138 bytes)
CREATE src/app/pages/secondepage/secondepage.page.spec.ts (726 bytes)
CREATE src/app/pages/secondepage/secondepage.page.ts (276 bytes)
CREATE src/app/pages/secondepage/secondepage.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (477 bytes)
[OK] Generated page!
```

Utilisation de **routerLink** pour naviguer et **ActivatedRoute**

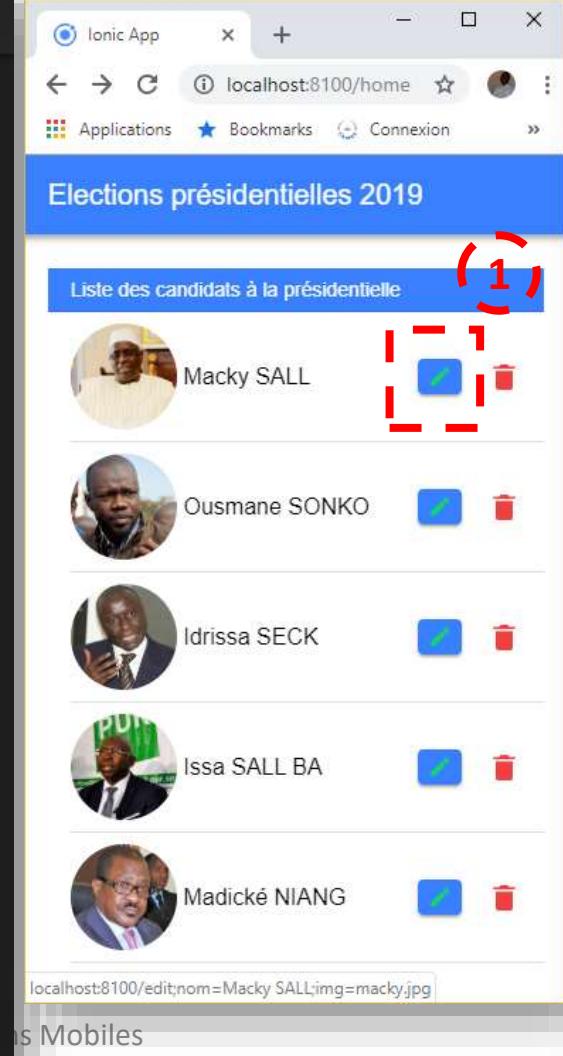
Pour récupérer les données.



# Les Composants de l'Interface Utilisateur: **ion-list**

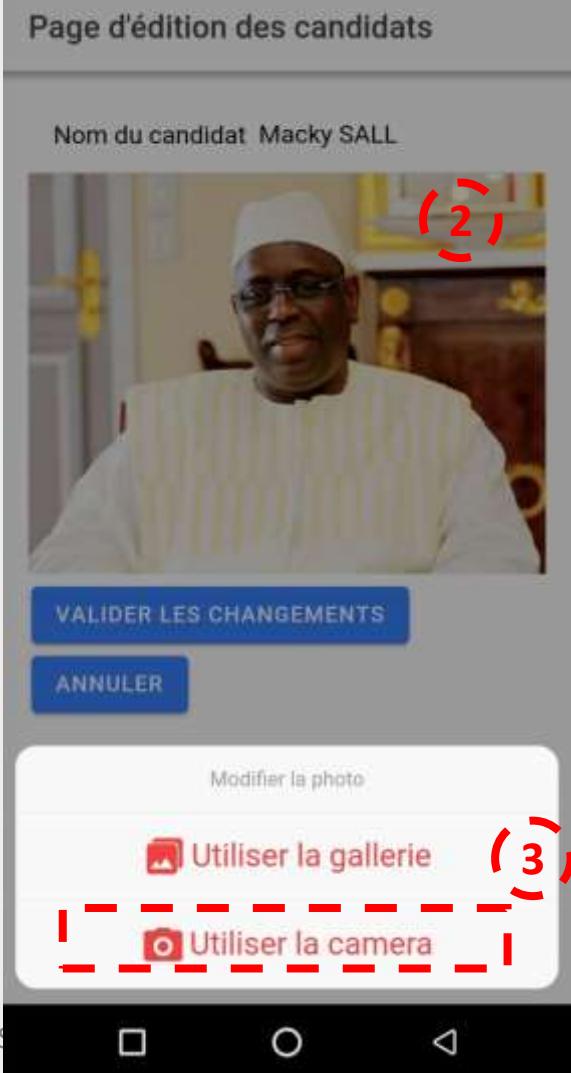
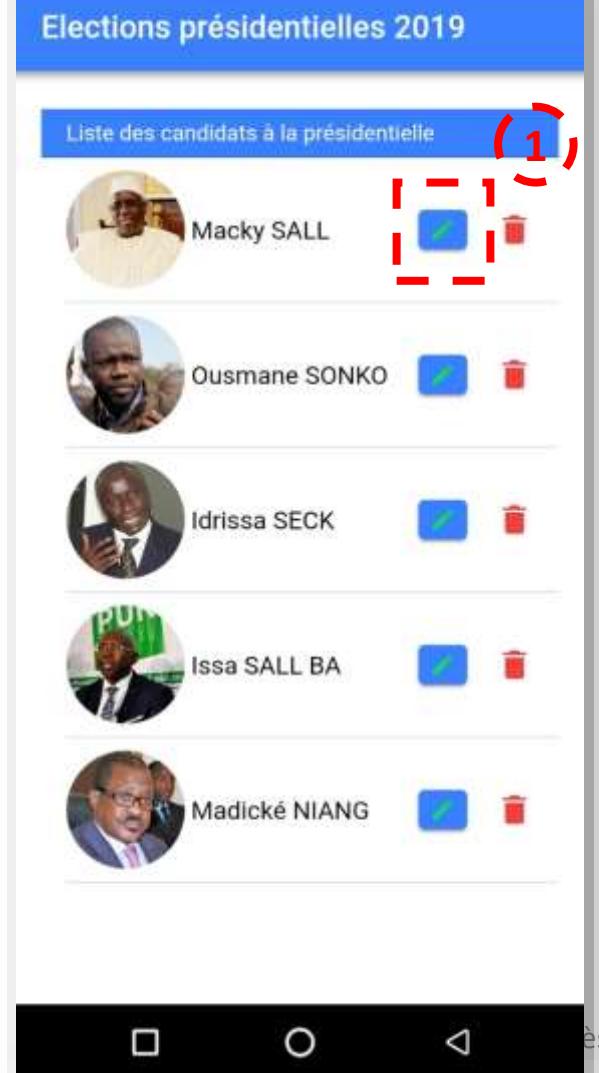
## Gestion de la modification et Camera

```
home.page.html ●  
9   <ion-content padding>  
10  <ion-list>  
11  
12    <ion-item-divider color="primary">  
13    | Liste des candidats à la présidentielle  
14    </ion-item-divider>  
15  
16    <ion-item *ngFor="let pers of candidats; let i=index;">  
17  
18      <ion-avatar style="width:75px; height: 75px;">  
19      |   
20      </ion-avatar>&nbsp;  
21      <ion-label>{{pers.nom}}</ion-label>  
22  
23      <ion-icon name="trash" color="danger" slot="end" (click)="supprimer(i)">  
24      </ion-icon>  
25  
26      <ion-button [routerLink]="[ '/edit', pers ]">  
27      | <ion-icon name="create" color="success"></ion-icon>  
28      </ion-button>  
29  
30    </ion-item>  
31  
32  </ion-list>  
33  
34</ion-content>
```



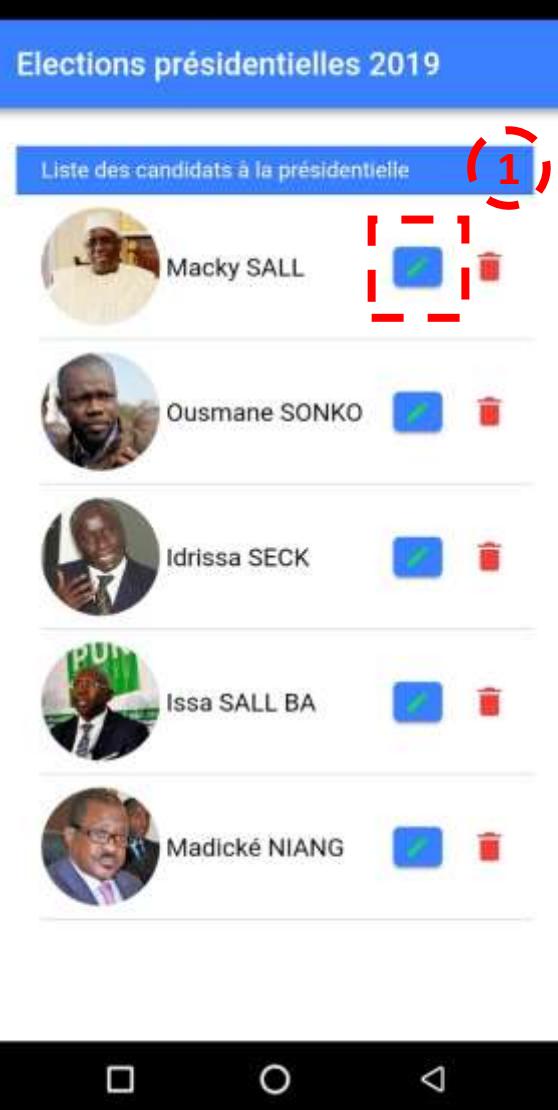
[Https://www.youtube.com/watch?v=6O2CRfZiSv4](https://www.youtube.com/watch?v=6O2CRfZiSv4)

Taper sur l'image pour faire apparaître l'**ActionSheet** permettant de choisir entre l'utilisation de la galerie ou de la caméra.



# Les Composants de l'Interface Utilisateur

## Gestion de la modification et Camera



Page d'édition des candidats

Nom du candidat Macky SALL



# Les Composants de l'Interface Utilisateur: **ion-list** Gestion de la modification et de la Camera

```
C:\Users\OSall\Desktop\AppIonic\projet1>ionic cordova plugin add cordova-plugin-camera  
> cordova plugin add cordova-plugin-camera --save  
Plugin "cordova-plugin-camera" already installed on android.  
Adding cordova-plugin-camera to package.json  
Saved plugin info for "cordova-plugin-camera" to config.xml
```

```
C:\Users\OSall\Desktop\AppIonic\projet1>npm install @ionic-native/camera@beta  
+ @ionic-native/camera@5.0.0  
updated 1 package and audited 51246 packages in 18.769s  
found 0 vulnerabilities
```

```
C:\Users\OSall\Desktop\AppIonic\projet1>npm i @ionic-native/file@beta  
+ @ionic-native/file@5.0.0  
updated 1 package and audited 51249 packages in 21.879s  
found 0 vulnerabilities
```



EXPLORATEUR

## ▶ ÉDITEURS OUVERTS

## ◀ PROJET1

- ▶ e2e
- ▶ node\_modules
- ▶ platforms
- ▶ plugins
- ▶ resources

## ◀ src

## ◀ app

## ◀ edit

TS edit.module.ts

▷ edit.page.html

∅ edit.page.scss

TS edit.page.spec.ts

TS edit.page.ts

## ◀ home

TS home.module.ts

▷ home.page.html

∅ home.page.scss

TS home.page.spec.ts

TS home.page.ts

## ▷ edit.page.html ✘

```
1  <ion-header>
2    <ion-toolbar>
3      <ion-title>Page d'édition des candidats</ion-title>
4    </ion-toolbar>
5  </ion-header>
6
7  <ion-content padding>
8
9    <ion-item>
10      <ion-label position="floating">Nom du candidat</ion-label>
11      <ion-input [(ngModel)]="pers.nom" value="{{pers.nom}}"></ion-input>
12    </ion-item>
13
14    <img [(ngModel)]="pers.img" [src]="imagePath" item-start (click)="selectionnerImage()">
15
16    <ion-button [routerLink]="[ '/home', pers ]">
17      Valider les changements
18    </ion-button>
19
20    <ion-button [routerLink]="[ '/home' ]">
21      Annuler
22    </ion-button>
23
24  </ion-content>
```

# Les Composants de l'Interface Utilisateur: **ion-list**

## Gestion des données

```
ts edit.page.ts ×

1  import { Component, OnInit } from '@angular/core';
2  import { ActivatedRoute } from '@angular/router';
3  import { Camera, CameraOptions } from '@ionic-native/camera/ngx';
4  import { ActionSheetController } from '@ionic/angular';
5  import { Personne } from '../home/home.page';
6
7  @Component({
8    selector: 'app-edit',
9    templateUrl: './edit.page.html',
10   styleUrls: ['./edit.page.scss'],
11 })
12
13 export class EditPage implements OnInit {
14
15   pers: Personne;
16   imagePath = '';
17
18   constructor(public actionSheetCtrl: ActionSheetController, public activatedRoute: ActivatedRoute,
19   public camera: Camera) {
20   }
21
22   ngOnInit() {
23     this.pers = new Personne(this.activatedRoute.snapshot.paramMap.get('nom'),
24     this.activatedRoute.snapshot.paramMap.get('img'));
25     this.imagePath = 'assets/' + this.pers.img;
26   }
}
```

# Les Composants de l'Interface Utilisateur: **ion-list**

## Gestion

```
28 |     async selectionnerImage() {
29 |       const actionSheet = await this.actionSheetCtrl.create({
30 |         header: 'Modifier la photo',
31 |         mode: 'ios',
32 |         buttons: [
33 |           {
34 |             text: 'Utiliser la galerie',
35 |             role: 'destructive',
36 |             icon: 'photos',
37 |             handler: () => {
38 |               this.changerImage(this.camera.PictureSourceType.PHOTOLIBRARY);
39 |             }
40 |           },
41 |           {
42 |             text: 'Utiliser la camera',
43 |             icon: 'camera',
44 |             role: 'destructive',
45 |             handler: () => {
46 |               this.changerImage(this.camera.PictureSourceType.CAMERA);
47 |             }
48 |           }
49 |         ]
50 |       });
51 |       await actionSheet.present();
52 |     }
```

# Les Composants de l'Interface

```
54     async changerImage(sourceTypeImg) {  
55         const options: CameraOptions = {  
56             quality: 100,  
57             destinationType: this.camera.DestinationType.DATA_URL,  
58             encodingType: this.camera.EncodingType.JPEG,  
59             mediaType: this.camera.MediaType.PICTURE,  
60             saveToPhotoAlbum: true,  
61             sourceType: sourceTypeImg  
62         };  
63  
64         this.camera.getPicture(options).then((imageData) => {  
65             this.imagePath = 'data:image/jpeg;base64,' + imageData;  
66         }, (err) => {  
67     });  
68         this.pers.img = this.imagePath;  
69     }  
70 }
```

## Aperçu de quelques options de la caméra:

- **sourceType**: récupère l'image depuis l'application photo native
- **destinationType**: retourne un URI pour l'image
- **quality**: maintenir 100% de la qualité d'origine de l'image
- **targetWidth / targetheight**: redimensionne l'image sur un maximum de 1000 pixels sur son côté le plus long - notez que cela ne modifie pas le rapport de format de l'image.
- **encodingType**: retourne l'image au format JPEG
- **correctOrientation**: faites pivoter l'image pour corriger l'orientation de l'appareil lorsque la photo a été prise



# Les Composants de l'Interface Utilisateur: listes de sélection **ion-select**, **ion-select-option**

- Les sélections sont des contrôles de formulaire pour sélectionner une ou plusieurs options parmi un ensemble d'options, similaires à un élément natif `<select>`. Lorsqu'un utilisateur appuie sur la sélection, une boîte de dialogue apparaît avec toutes les options dans une grande liste facile à sélectionner.
- Une sélection doit être utilisée avec les éléments enfants **<ion-select-option>**. Si un attribut `value` n'est pas attribué à l'option enfant, son texte sera utilisé comme valeur.
- Si la valeur est définie sur la **<ion-select>**, l'option sélectionnée sera choisie en fonction de cette valeur. Sinon, l'attribut sélectionné peut être utilisé sur **<ion-select-option>**.

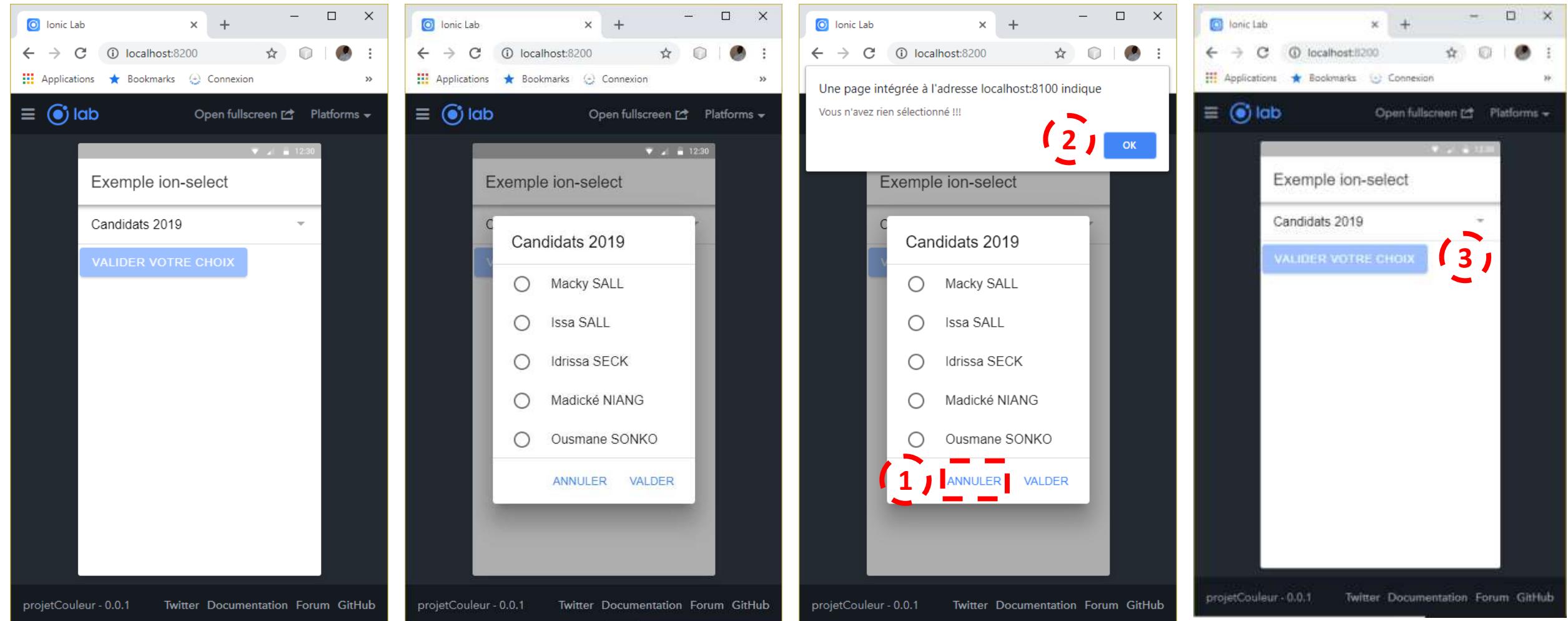
# Les Composants de l'Interface Utilisateur: listes de sélection **ion-select**, **ion-select-option**

- Attributs:

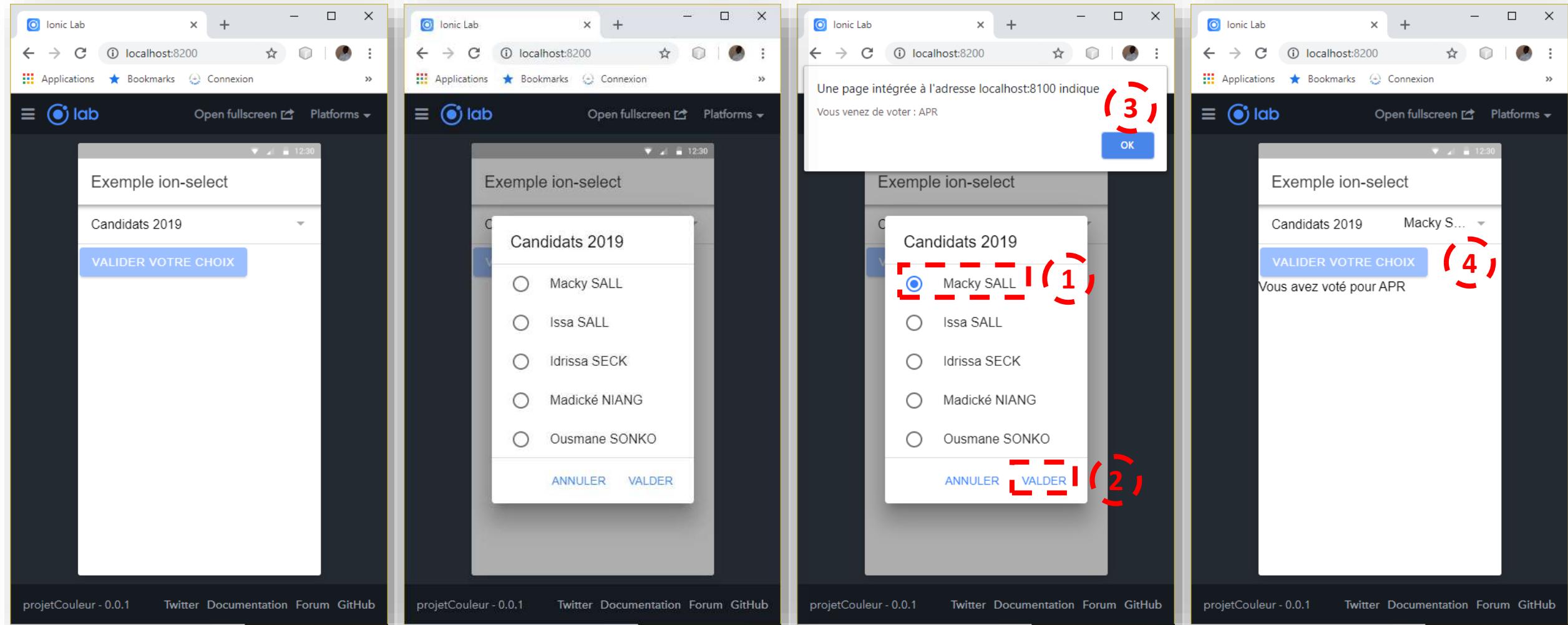
- **Multiple** : En ajoutant l'attribut multiple à sélectionner, les utilisateurs peuvent sélectionner plusieurs options. Remarque: les interfaces Action-Sheet et Popover ne fonctionneront pas avec une sélection multiple.
- Par défaut deux boutons de sélections `ok-text="Okay" cancel-text="Annuler"`
- **disabled** un attribut boolean
- **name**: nom du contrôleur
- **Placeholder**: texte affiché si le select est vide
- **selectedText**: Le texte à afficher à la place de la valeur de l'option sélectionnée.

Events	
Name	Description
ionBlur	Emitted when the select loses focus.
ionCancel	Emitted when the selection is cancelled.
ionChange	Emitted when the value has changed.
ionFocus	Emitted when the select has focus.

# Les Composants de l'Interface Utilisateur: listes de sélection **ion-select**, **ion-select-option**



# Les Composants de l'Interface Utilisateur: listes de sélection **ion-select**, **ion-select-option**





home.page.html

```
8 <ion-content>
9   <ion-item>
10    <ion-label>Candidats 2019</ion-label>
11    <ion-select multiple="false" [(ngModel)]="donnees"
12      ok-text="Valider" cancel-text="Annuler"
13      (ionChange)="onChange($event, true)"
14      (ionCancel)="onChange($event, false)">
15      <ion-select-option value="APR">
16        Macky SALL
17      </ion-select-option>
18      <ion-select-option value="PUR">
19        Issa SALL
20      </ion-select-option>
21      <ion-select-option value="Rewmi">
22        Idrissa SECK
23      </ion-select-option>
24      <ion-select-option value="Coalition Madické 2019">
25        Madické NIANG
26      </ion-select-option>
27      <ion-select-option value="PASTEF">
28        Ousmane SONKO
29      </ion-select-option>
30    </ion-select>
31  </ion-item>
32
33  <ion-button (click)="validerChoix()" disabled="{{activerBouton}}">
34    Valider votre choix
35  </ion-button><br>
36  <ion-label>
37    {{donnees == null ? '' : 'Vous avez voté pour '+donnees}}
38  </ion-label>
39
40  </ion-content>
```

home.page.ts

```
1 import { Component } from '@angular/core';
2 import { ModalController, IonSelectOption } from '@ionic/angular';
3 import { PageModalePage } from '../page-modale/page-modale.page';
4 import { OverlayEventDetail } from '@ionic/core';
5
6 @Component({
7   selector: 'app-home',
8   templateUrl: 'home.page.html',
9   styleUrls: ['home.page.scss'],
10 })
11 export class HomePage {
12   donnees = null;
13   desactiverBouton = true;
14   constructor() {
15   }
16
17   async onChange($event: any, bool: boolean) {
18     if (bool && this.donnees != null) {
19       this.desactiverBouton = false;
20       await alert('Vous venez de voter : '
21         + $event.target.value);
22     } else {
23       this.desactiverBouton = true;
24       await alert('Vous n\'avez rien sélectionné !!! ');
25     }
26   }
27
28   validerChoix() {
29     if (this.donnees != null) {
30       alert(this.donnees);
31       this.donnees = null;
32     }
33   }
34
35 }
36
```

Les Composants de l'Interface Utilisateur: **Grid:**  
**ion-col, ion-row, ion-grid, Infinite Scroll, ion-infinite-scroll-content, ion-infinite-scroll**

# Les éléments modaux



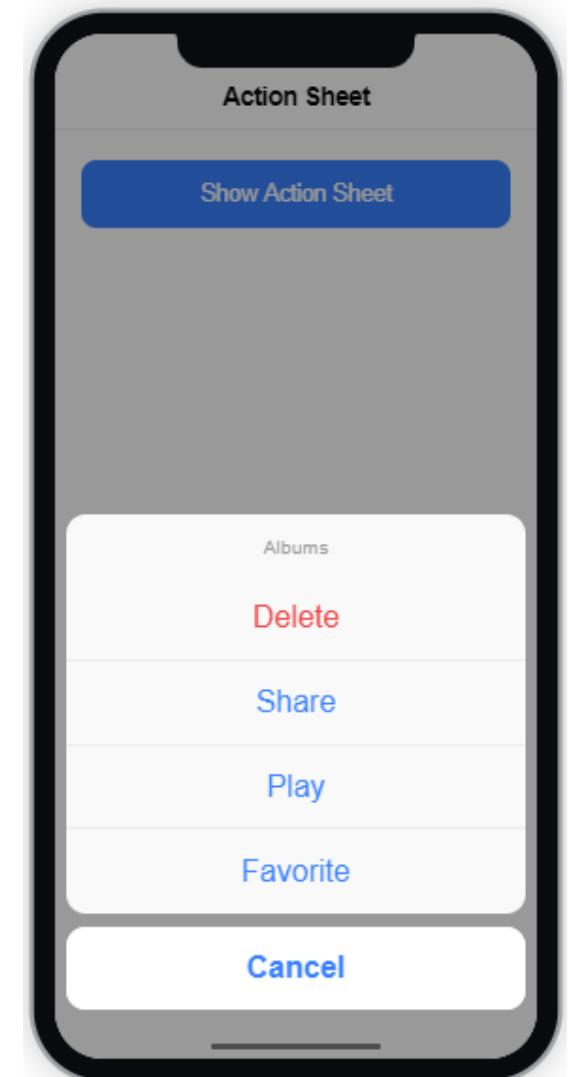
# Les éléments modaux

- Destiné à une saisie rapide ou à des alertes
- Apparaître sur le contenu principal de l'application
- Deux styles différents
  - Boîtes de dialogues modales
  - pages modales
- Généralement déclenché dans le modèle ou le contrôleur (.ts) plutôt que dans la vue (.html)

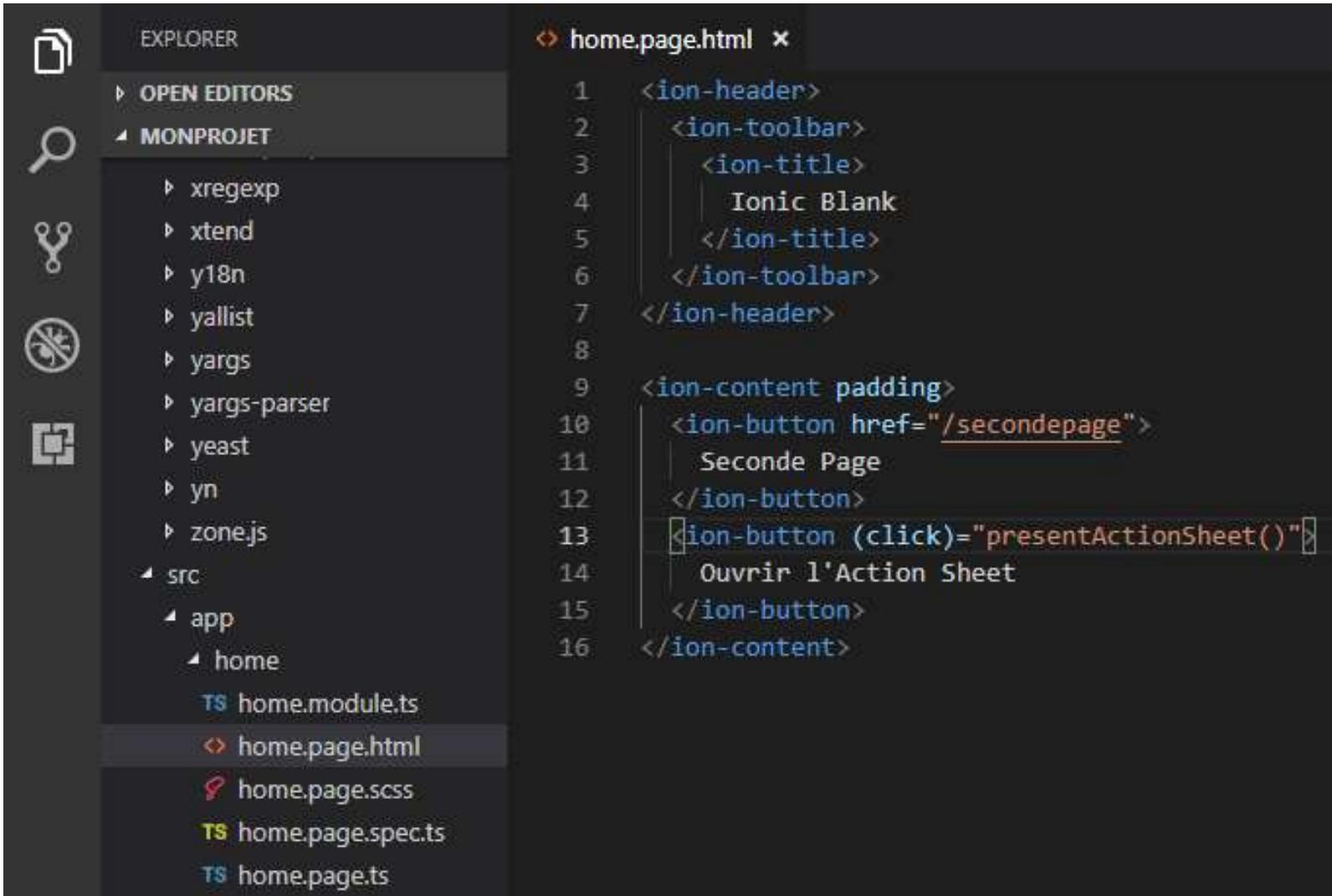
# Les Composants de l'Interface Utilisateur: **ion-action-sheet** et **ion-action-sheet-controller**

[Https://ionicframework.com/docs/api/action-sheet](https://ionicframework.com/docs/api/action-sheet)

- Une action sheet est une boîte de dialogue qui affiche un ensemble d'options. Il apparaît en haut du contenu de l'application et doit être rejeté manuellement par l'utilisateur avant qu'il ne puisse reprendre son interaction avec l'application.
- Les options destructives sont mises en évidence en mode iOS. Il existe plusieurs façons de supprimer la feuille d'action, notamment en tapant sur la toile de fond ou en appuyant sur la touche Échap.



# Les Composants de l'Interface Utilisateur: Action Sheet



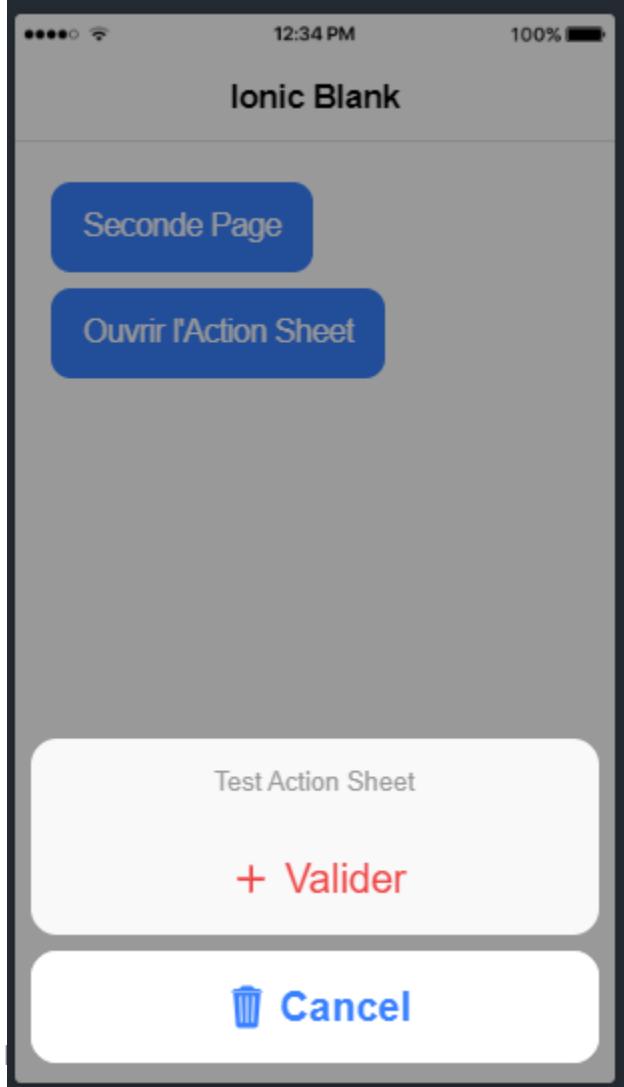
The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar:
  - OPEN EDITORS
  - MONPROJET (selected)
  - xregexp
  - xtend
  - y18n
  - yallist
  - yargs
  - yargs-parser
  - yeast
  - yn
  - zone.js
  - src
  - app
    - home
      - home.module.ts
      - home.page.html (selected)
      - home.page.scss
      - home.page.spec.ts
      - home.page.ts
- home.page.html** file content:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  <ion-button href="/secondepage">
11    Seconde Page
12  </ion-button>
13  <ion-button (click)="presentActionSheet()">
14    Ouvrir l'Action Sheet
15  </ion-button>
16 </ion-content>
```

# Les Composants

## Action Sheet



EXPLORER

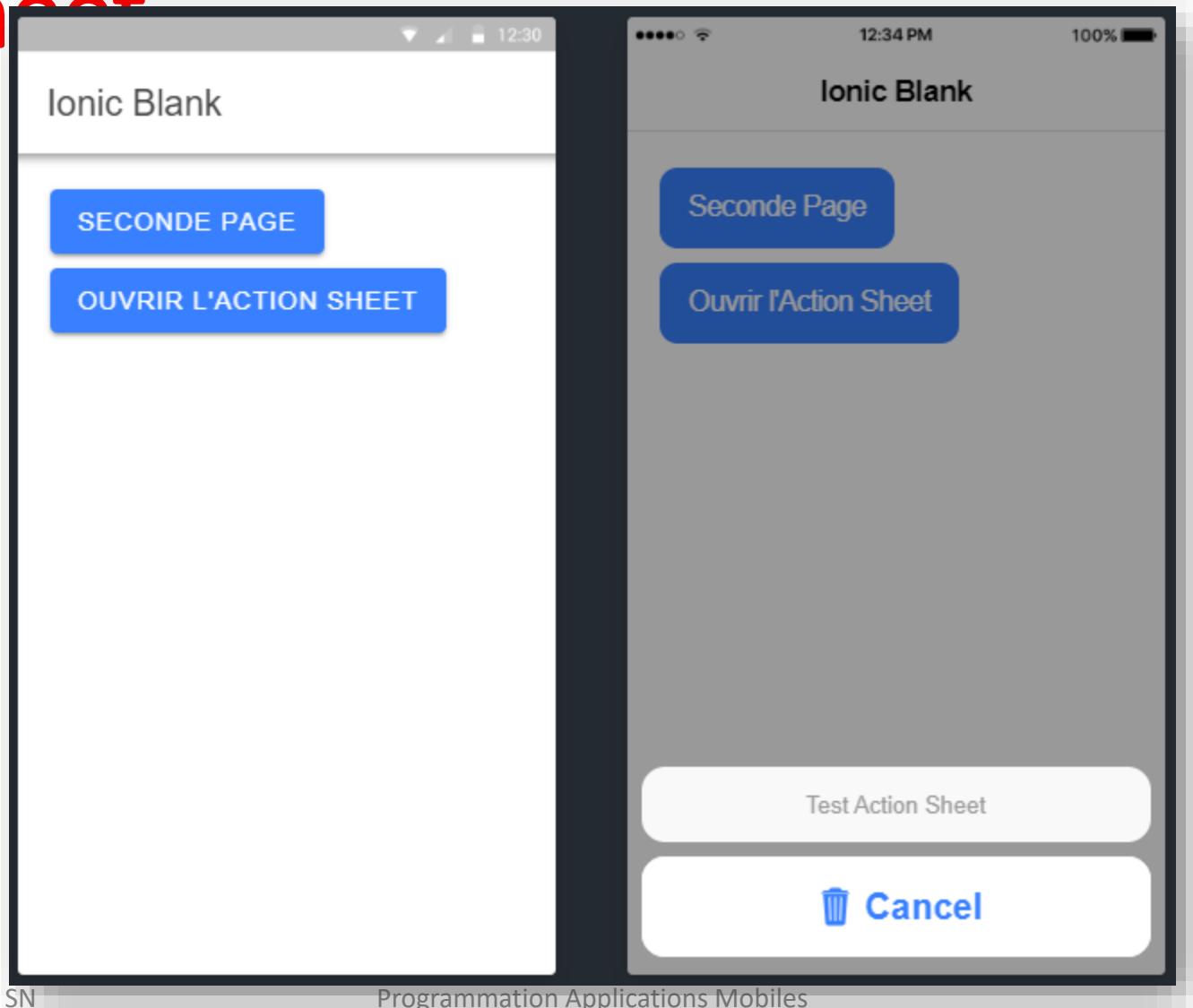
- OPEN EDITORS
- MONPROJET
  - xregexp
  - xtend
  - y18n
  - yallist
  - yargs
  - yargs-parser
  - yeast
  - yn
  - zone.js
- SRC
  - app
    - home
      - TS home.module.ts
      - home.page.html
      - home.page.scss
      - TS home.page.spec.ts
      - TS home.page.ts
    - secondepage
      - TS secondepage.module.ts
      - secondepage.page.html
      - secondepage.page.scss
      - TS secondepage.page.sp...
      - TS secondepage.page.ts
  - app-routing.module.ts
  - app.component.html
  - TS app.component.spec.ts
  - TS app.component.ts

TS home.page.ts ×

```
1 import { Component } from '@angular/core';
2 import { ActionSheetController } from '@ionic/angular';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   constructor(public actionsheet: ActionSheetController) { }
11   async presentActionSheet() {
12     const actionsheet = await this.actionsheet.create({
13       header: "Test Action Sheet",
14       mode: 'ios',
15       buttons: [
16         {
17           text: 'Cancel',
18           role: 'cancel',
19           icon: 'trash',
20           handler: () => {
21             console.log("Vous m'avez cliqué");
22           }
23         },
24         {
25           text: 'Valider',
26           role: 'destructive',
27           icon: 'add',
28           handler: () => {
29             console.log("Vous m'avez cliqué");
30           }
31         }
32       ]);
33     await actionsheet.present();
34   }
35 }
```

OUTLINE

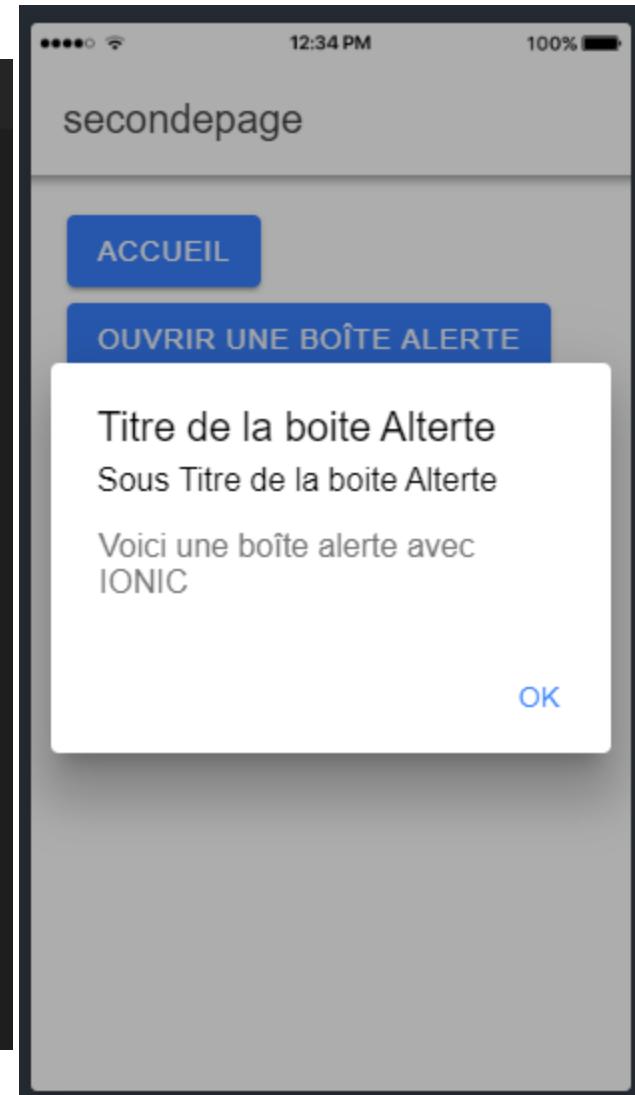
# Les Composants de l'Interface Utilisateur: Action Sheet



# Les Composants de l'Interface Utilisateur: Alert

```
secondepage.page.html ×
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>secondepage</ion-title>
4   </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8   <ion-button href="/home">
9     Accueil
10    </ion-button>
11    <ion-button (click)="lancerAlerte()">
12      Ouvrir une boîte alerte
13    </ion-button>
14 </ion-content>
```

```
ts secondepage.page.ts ×
1 import { Component, OnInit } from '@angular/core';
2 import { AlertController } from '@ionic/angular';
3
4 @Component({
5   selector: 'app-secondepage',
6   templateUrl: './secondepage.page.html',
7   styleUrls: ['./secondepage.page.scss'],
8 })
9 export class SecondepagePage implements OnInit {
10
11   constructor(private alerteControleur: AlertController) { }
12
13   async ngOnInit() {
14   }
15   async lancerAlerte() {
16     const alerte = await this.alerteControleur.create({
17       header: "Titre de la boîte Alterte",
18       subHeader: "Sous Titre de la boîte Alterte",
19       message: "Voici une boîte alerte avec IONIC",
20       buttons: [ 'OK' ]
21     });
22     await alerte.present();
23   }
24 }
```



secondepage.page.ts - monProjet - Visual Studio Code

File Edit Selection View Go Debug Terminal Help

EXPLORER

OPEN EDITORS

MONPROJET

- src
  - app
    - home
      - home.module.ts
      - home.page.html
      - home.page.scss
      - home.page.spec.ts
      - home.page.ts
    - secondepage
      - secondepage.module.ts
      - secondepage.page.html
      - secondepage.page.scss
      - secondepage.page.sp...
      - secondepage.page.ts
  - app-routing.module.ts
  - app.component.html
  - app.components.spec.ts
  - app.components.ts
  - app.module.ts
  - assets
  - environments
  - theme
  - global.scss
  - index.html
  - karma.conf.js
  - main.ts

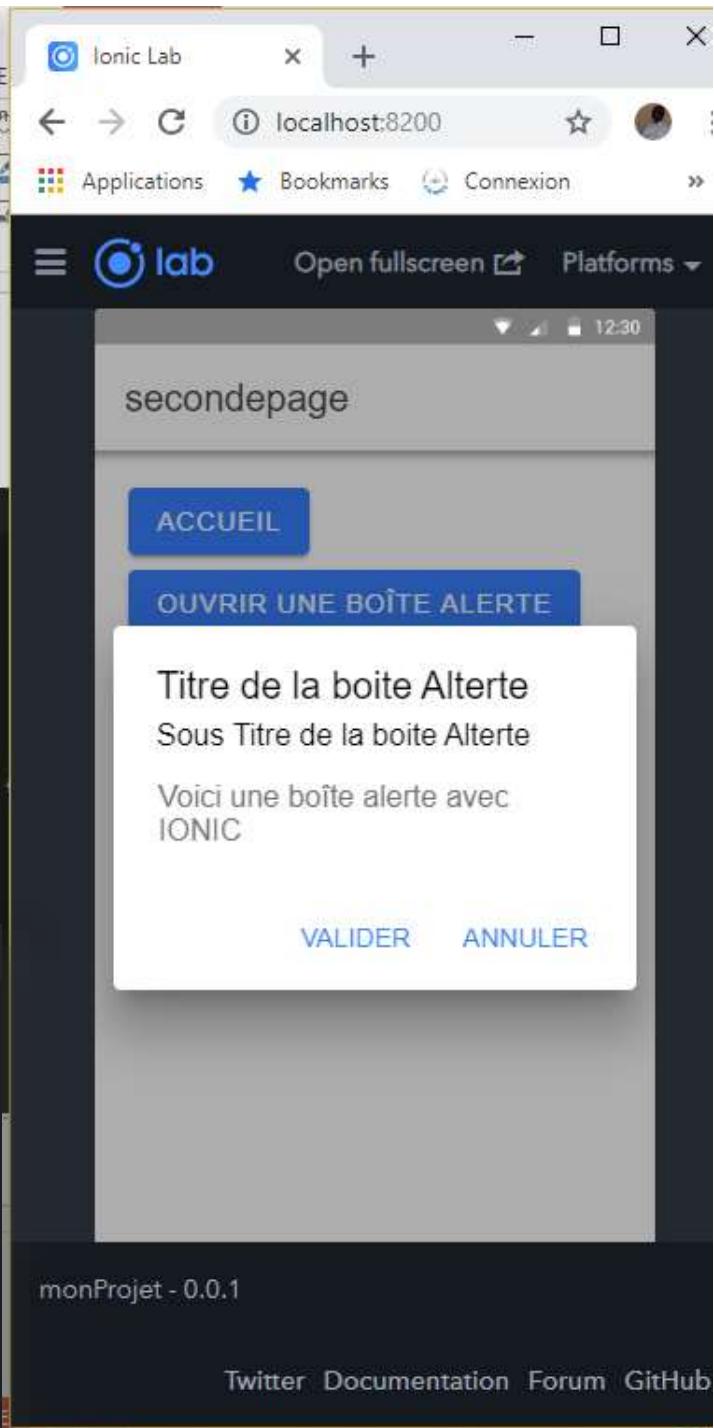
OUTLINE

0 ▲ 0

```
4  @Component({
5    selector: 'app-secondepage',
6    templateUrl: './secondepage.page.html',
7    styleUrls: ['./secondepage.page.scss'],
8  })
9  export class SecondepagePage implements OnInit {
10
11    constructor(private alerteControleur: AlertController) { }
12
13    async ngOnInit() {
14    }
15    async lancerAlerte() {
16      const alerte = await this.alerteControleur.create({
17        header: "Titre de la boite Alterte",
18        subHeader: "Sous Titre de la boite Alterte",
19        message: "Voici une boîte alerte avec IONIC",
20        buttons: [
21          {
22            text: "Valider",
23            handler: () => {
24              console.log("Click sur Valider");
25            }
26          },
27          {
28            text: "Annuler",
29            role: "cancel",
30            handler: () => {
31              console.log("Annuler");
32            }
33          }
34        ]
35      });
36      await alerte.present();
37    }
38  }
```

secondepage.page.html

```
1  <ion-header>
2  <ion-toolbar>
3  | <ion-title>secondepage</ion-title>
4  </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8   <ion-button href="/home">
9     Accueil
10    </ion-button>
11   <ion-button (click)="lancerAlerte()">
12     Ouvrir une boîte alerte
13   </ion-button>
14 </ion-content>
```



# Les Composants de l'Interface Utilisateur: **ion-modal** et **ion-modal-controller**

- Un **ion-modal** est une boîte de dialogue qui apparaît au-dessus du contenu de l'application et doit être ignorée par l'application avant que l'interaction ne puisse reprendre.
- Il est utile en tant que composant de sélection lorsqu'il existe de nombreuses options, lors du filtrage d'éléments dans une liste, ainsi que dans de nombreux autres cas d'utilisation.
- Les contrôleurs **ion-modal-controller** contrôlent par programmation le composant modal. Des modaux peuvent être créés et exclus du contrôleur modal.

# Les Composants de l'Interface Utilisateur: **ion-modal** et **ion-modal-controller**

## Properties

**animated**

**backdropDismiss**

**componentProps**

Description	If <code>true</code> , the modal will animate. <th>Description</th> <td>If <code>true</code>, the modal will be dismissed when the backdrop is clicked.</td> <th>Description</th> <td>The data to pass to the modal component.</td>	Description	If <code>true</code> , the modal will be dismissed when the backdrop is clicked.	Description	The data to pass to the modal component.
Attribute	<code>animated</code>	Attribute	<code>backdrop-dismiss</code>	Type	<code>undefined   { [key: string]: any; }</code>
Type	<code>boolean</code> <th>Type</th> <td><code>boolean</code><td></td><td></td></td>	Type	<code>boolean</code> <td></td> <td></td>		

**component**

**cssClass**

Description	The component to display inside of the modal.	Description	Additional classes to apply for custom CSS. If multiple classes are provided they should be separated by spaces.
Attribute	<code>component</code>	Attribute	<code>css-class</code>
Type	<code>Function   HTMLElement   null   string</code>	Type	<code>string   string[]   undefined</code>

# Les Composants de l'Interface Utilisateur: **ion-modal** et **ion-modal-controller**

## Properties

### enterAnimation

Description	Animation to use when the modal is presented.
-------------	---

Type	((Animation: Animation, baseEl: any, opts?: any) => Promise<Animation>)   undefined
------	---

### keyboardClose

Description	If <code>true</code> , the keyboard will be automatically dismissed when the overlay is presented.
-------------	--

Attribute	keyboard-close
-----------	----------------

Type	boolean
------	---------

### leaveAnimation

Description	Animation to use when the modal is dismissed.
-------------	---

Type	((Animation: Animation, baseEl: any, opts?: any) => Promise<Animation>)   undefined
------	---

### mode

Description	The mode determines which platform styles to use.
-------------	---

Attribute	mode
-----------	------

Type	"ios"   "md"
------	--------------

# Les Composants de l'Interface Utilisateur: ion-

Properties **ion-modal** et **ion-modal-controller**

showBackdrop

Description	If <code>true</code> , a backdrop will be displayed behind the modal.
Attribute	<code>show-backdrop</code>
Type	<code>boolean</code>

## Events

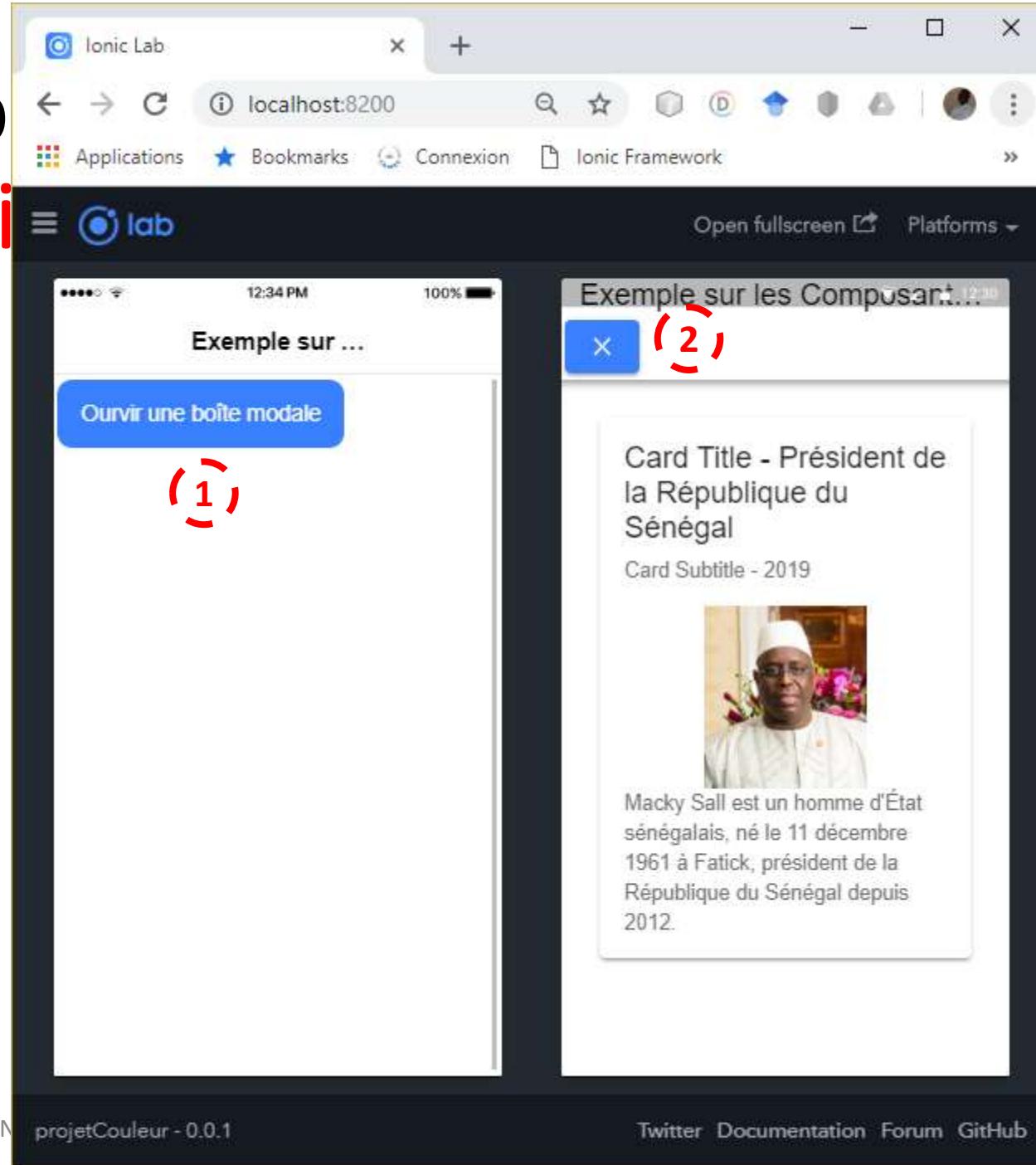
Name	Description
<code>ionModalDidDismiss</code>	Emitted after the modal has dismissed.
<code>ionModalDidPresent</code>	Emitted after the modal has presented.
<code>ionModalWillDismiss</code>	Emitted before the modal has dismissed.
<code>ionModalWillPresent</code>	Emitted before the modal has presented.

## CSS Custom Properties

Name	Description
<code>--background</code>	Background of the modal content
<code>--border-color</code>	Border color of the modal content
<code>--border-radius</code>	Border radius of the modal content
<code>--border-style</code>	Border style of the modal content
<code>--border-width</code>	Border width of the modal content
<code>--height</code>	Height of the modal
<code>--max-height</code>	Maximum height of the modal
<code>--max-width</code>	Maximum width of the modal
<code>--min-height</code>	Minimum height of the modal
<code>--min-width</code>	Minimum width of the modal
<code>--width</code>	Width of the modal

# Les Composants modal et ion- modal

ateur: ion-



# Les Composants de l'Interface Utilisateur : modal et ion-modal-controller

- Créer une nouvelle page modale nommée PageModale

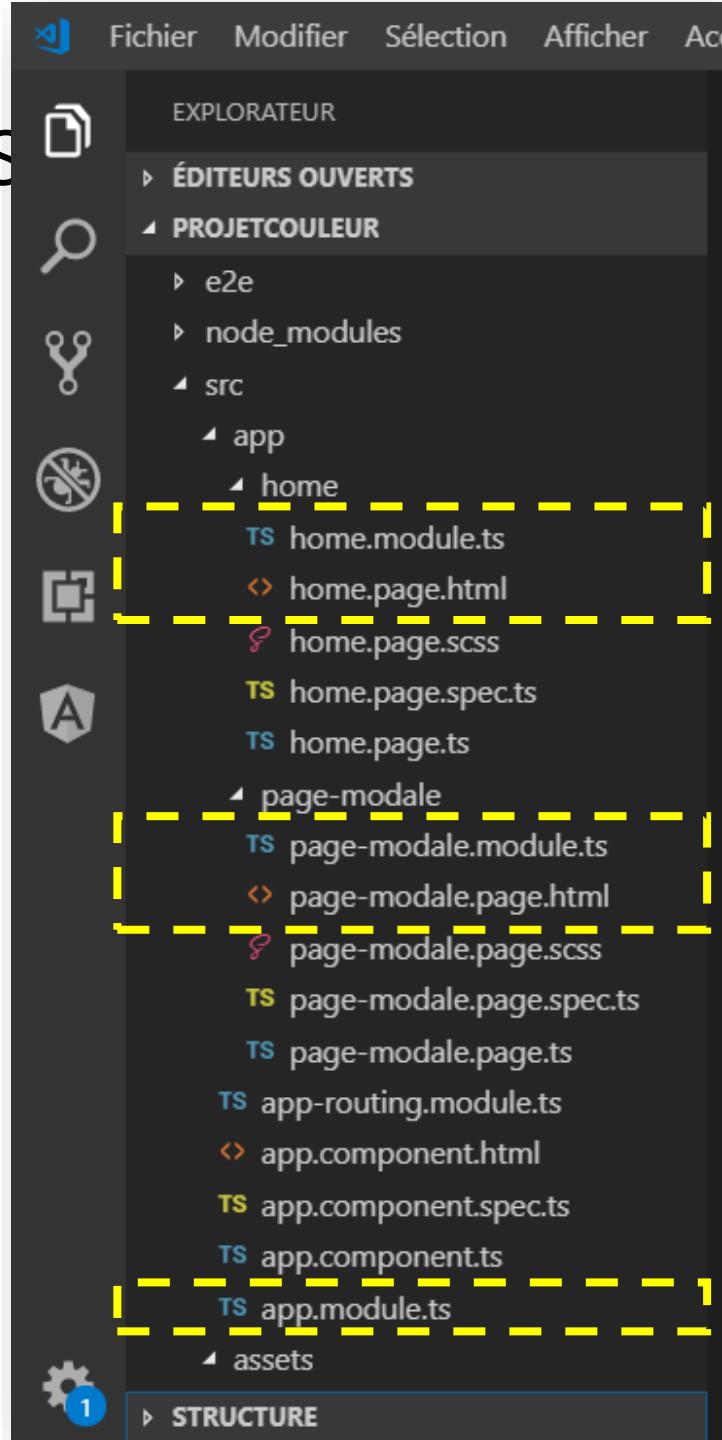
```

PROBLÈMES 1 SORTIE CONSOLE DE DÉBOGAGE TERMINAL 1: po

Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

PS C:\Users\OSall\Desktop\AppIonic\projetCouleur> ionic g page PageModale
> ng generate page PageModale
CREATE src/app/page-modale/page-modale.module.ts (564 bytes)
CREATE src/app/page-modale/page-modale.page.html (137 bytes)
CREATE src/app/page-modale/page-modale.page.spec.ts (720 bytes)
CREATE src/app/page-modale/page-modale.page.ts (275 bytes)
CREATE src/app/page-modale/page-modale.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (470 bytes)
[OK] Generated page!
PS C:\Users\OSall\Desktop\AppIonic\projetCouleur>

```



# Les Composants de l'Interface Utilisateur: **ion-modal** et **ion-modal-controller**

The screenshot shows the Visual Studio Code interface with the following details:

- Toolbar:** Fichier, Modifier, Sélection, Afficher, Accéder, Déboguer, Terminal, Aide.
- File Explorer:** Shows files in the 'PROJETCOULEUR' folder:
  - page-modale.page.html
  - page-modale.page.scss
  - page-modale.page.spec.ts
  - page-modale.page.ts
  - app-routing.module.ts
  - app.component.html
  - app.component.spec.ts
  - app.component.ts
  - app.module.ts
- Code Editor:** The file 'app.module.ts' is open, displaying the following TypeScript code:

```
7 import { StatusBar } from '@ionic-native/status-bar/ngx';
8
9 import { AppComponent } from './app.component';
10 import { AppRoutingModule } from './app-routing.module';
11 import { PageModalePage } from './page-modale/page-modale.page';
12
13
14 @NgModule({
15   declarations: [AppComponent, PageModalePage],
16   entryComponents: [PageModalePage],
17   imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule],
18   providers: [
19     StatusBar,
20     SplashScreen,
21     { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
22   ],
23   bootstrap: [AppComponent]
24 })
25 export class AppModule {}
```
- Status Bar:** Prof. Ousmane SALL, Univ. Thiès, SN | Programmation Applications Mobiles | 670

# Les Composants de l'Interface Utilisateur

Fichier Modifier Sélection Afficher Accéder Déboguer Terminal Aide home.page.ts - projetCouleur - Visual Studio Code

home.page.html

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Exemple sur les Composants
5       de l'Interface Utilisateur: ion-modal
6       et ion-modal-controller
7     </ion-title>
8   </ion-toolbar>
9 </ion-header>
10
11 <ion-content>
12   <ion-button (click)="ouvrirBoiteModale()">
13     Ouvrir une boîte modale
14   </ion-button>
15 </ion-content>
```

home.page.ts

```
1 import { Component } from '@angular/core';
2 import { ModalController } from '@ionic/angular';
3 import { PageModalePage } from '../page-modale/page-modale.page';
4
5 @Component({
6   selector: 'app-home',
7   templateUrl: 'home.page.html',
8   styleUrls: ['home.page.scss'],
9 })
10 export class HomePage {
11   constructor(private modalController: ModalController) {
12   }
13 }
14
15 async ouvrirBoiteModale() {
16   const modal = await this.modalController.create({
17     component: PageModalePage,
18     componentProps: { value: 123 }
19   });
20   return await modal.present();
21 }
```



page-modale.page.html

```
1  <ion-header>
2    <ion-title>
3      Exemple sur les Composants de l'Interface Utilisateur
4    </ion-title>
5    <ion-button (click)="fermerBoiteModale()" slot="icon-only">
6      <ion-icon name="close"></ion-icon>
7    </ion-button>
8  </ion-header>
9
10 <ion-content padding>
11   <ion-card>
12     <ion-card-header>
13       <ion-card-title>Card Title - Président de
14         la République du Sénégal
15       </ion-card-title>
16       <ion-card-subtitle>
17         Card Subtitle - 2019
18       </ion-card-subtitle>
19     </ion-card-header>
20
21     <ion-card-content>
22       
23       Macky Sall est un homme d'État sénégalais,
24       président de la République du Sénégal depuis
25       2012.
26     </ion-card-content>
27   </ion-card>
28 </ion-content>
```



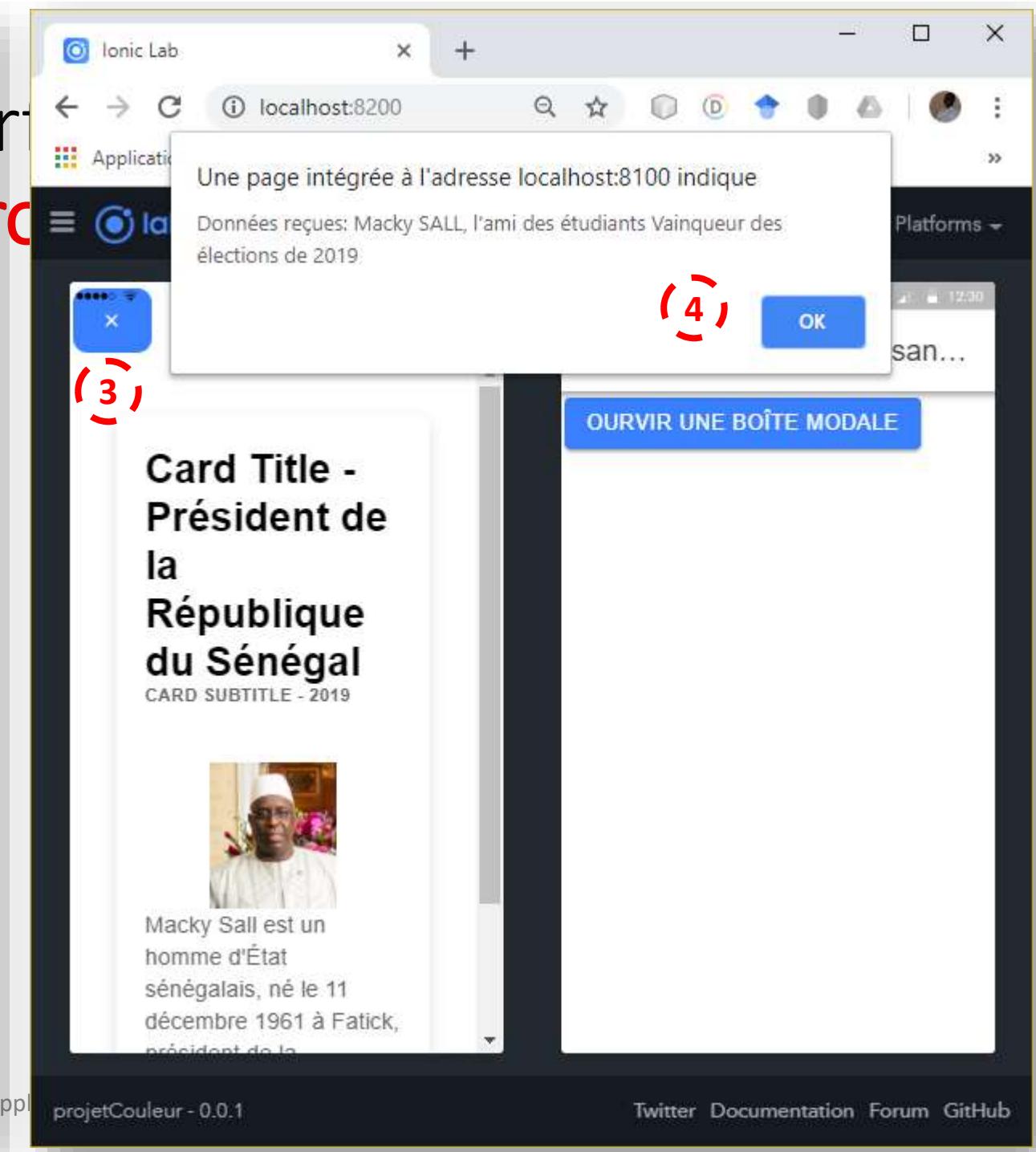
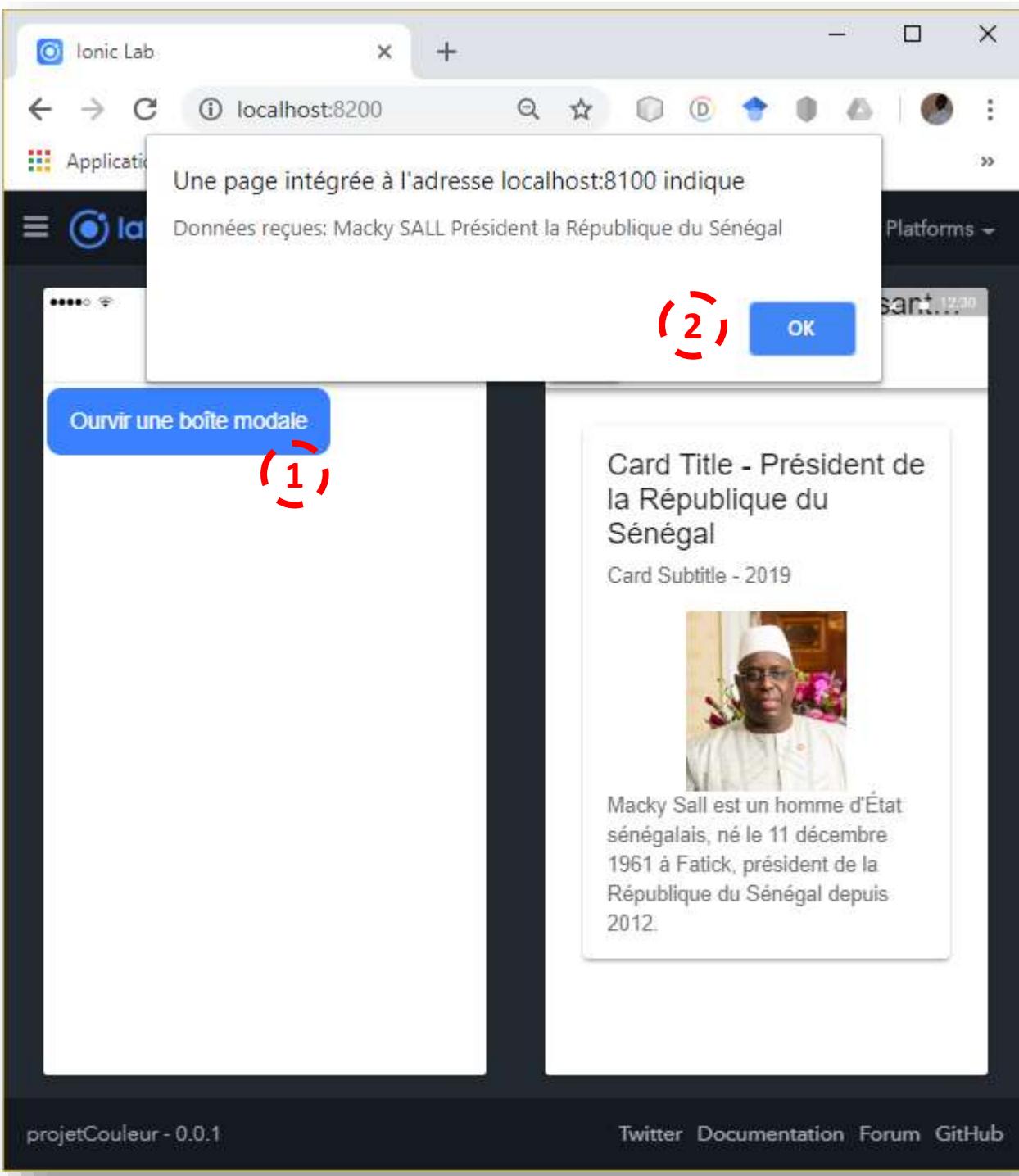
page-modale.page.ts

```
1  import { Component, OnInit } from '@angular/core';
2  import { ModalController } from '@ionic/angular';
3
4  @Component({
5    selector: 'app-page-modale',
6    templateUrl: './page-modale.page.html',
7    styleUrls: ['./page-modale.page.scss'],
8  })
9  export class PageModalePage implements OnInit {
10
11  constructor(private modalController: ModalController) { }
12
13  ngOnInit() {
14  }
15
16
17  fermerBoiteModale() {
18    this.modalController.dismiss();
19  }
20
21 }
22
```

# Les Composants de l'Interface Utilisateur: **ion-modal** et **ion-modal-controller**: passage de données

- Comme dans Ionic v3, la classe **ModalController** est utilisée pour créer et interagir avec un modal.
- Ce contrôleur offre une méthode de création qui nous permettra de transmettre des paramètres comme spécifiés dans l'exemple suivant avec **composantProps** et un gestionnaire d'événement **onDidDismiss** qui nous permettra d'écouter l'action de fermeture du modal afin d'obtenir le résultat.

<https://ionicframework.com/docs/api/modal>





EXPLORATEUR

- ÉDITEURS OUVERTS
- PROJETCOULEUR
  - e2e
  - node\_modules
  - src
    - app
      - home
        - TS home.module.ts
        - ▷ home.page.html
        - ↗ home.page.scss
        - TS home.page.spec.ts
        - TS home.page.ts
      - page-modale
        - TS page-modale.module.ts
        - ▷ page-modale.page.html
        - ↗ page-modale.page.scss
        - TS page-modale.page.spec.ts
        - TS page-modale.page.ts
    - TS app-routing.module.ts
    - ▷ app.component.html
    - TS app.component.spec.ts
    - TS app.component.ts
    - TS app.module.ts
    - assets



STRUCTURE

TS home.page.ts ×

```
7   selector: 'app-home',
8   templateUrl: 'home.page.html',
9   styleUrls: ['home.page.scss'],
10 }
11 export class HomePage {
12   constructor(private modalController: ModalController) {
13   }
14
15   async ouvrirBoiteModale() {
16     let mesDonnees = {
17       nom: 'Macky SALL',
18       fonction: 'Président la République du Sénégal'
19     };
20     const modal = await this.modalController.create({
21       component: PageModalePage,
22       componentProps: { value: mesDonnees }
23     );
24
25     modal.onDidDismiss().then((donneesRecues: OverlayEventDetail) => {
26       if (donneesRecues !== null) {
27         mesDonnees = donneesRecues.data;
28         alert('Données reçues: ' + mesDonnees.nom + ' ' + mesDonnees.fonction);
29       }
30     );
31   }
32
33   return await modal.present();
34 }
35 }
```



EXPLORATEUR

- ▷ ÉDITEURS OUVERTS
- ▲ PROJETCOULEUR
  - ▷ e2e
  - ▷ node\_modules
  - ▲ src
    - ▲ app
      - ▷ home
      - ▲ page-modale
        - TS page-modale.module.ts
        - ▷ page-modale.page.html
        - 🔗 page-modale.page.scss
        - TS page-modale.page.spec.ts
        - TS page-modale.page.ts
    - TS app-routing.module.ts
    - ▷ app.component.html
    - TS app.component.spec.ts
    - TS app.component.ts
    - TS app.module.ts
    - ▲ assets
      - ▷ icon
      - 🖼️ Macky\_Sall.jpg
      - SVG shapes.svg
    - ▷ environments
    - ▷ theme

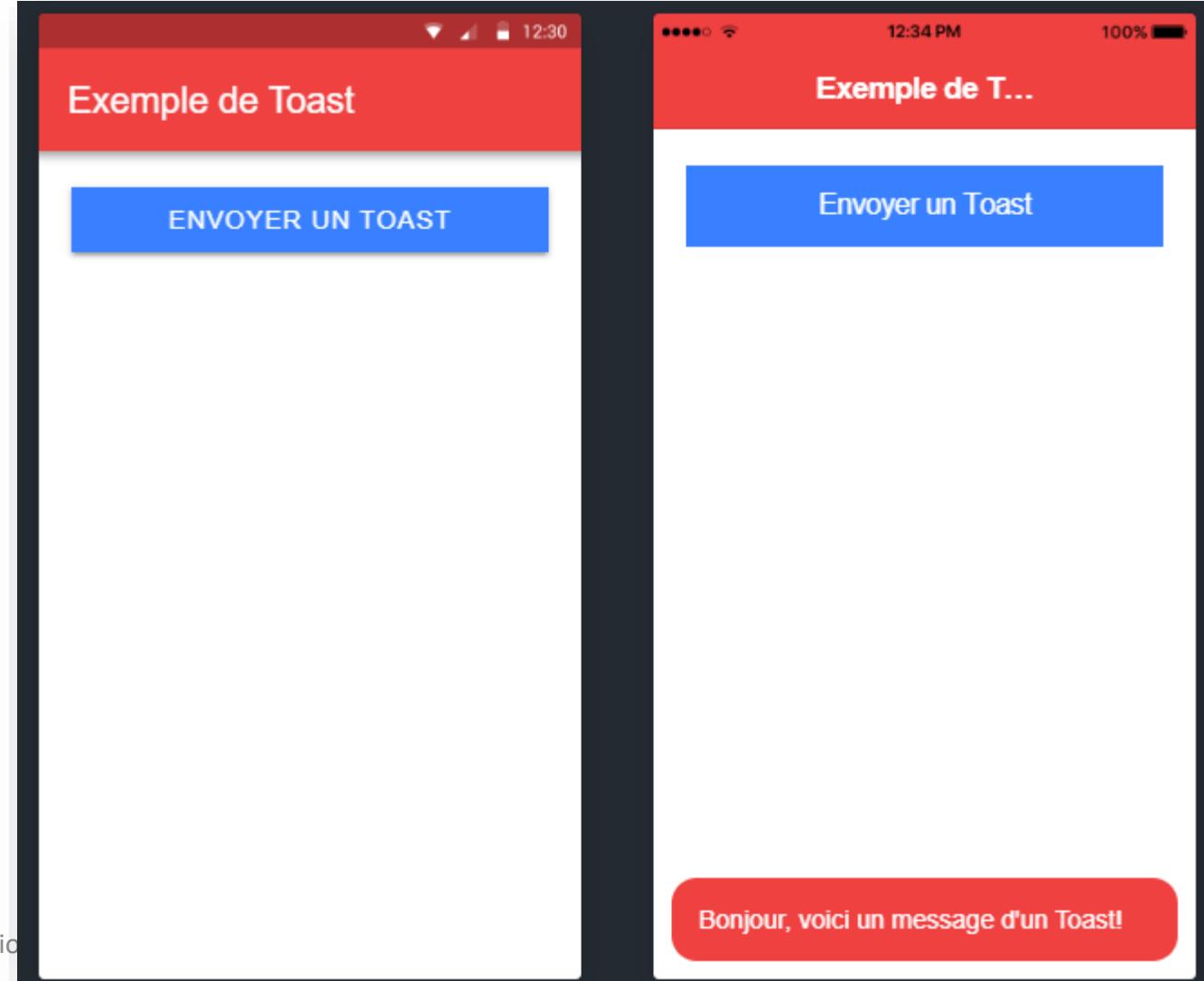


STRUCTURE

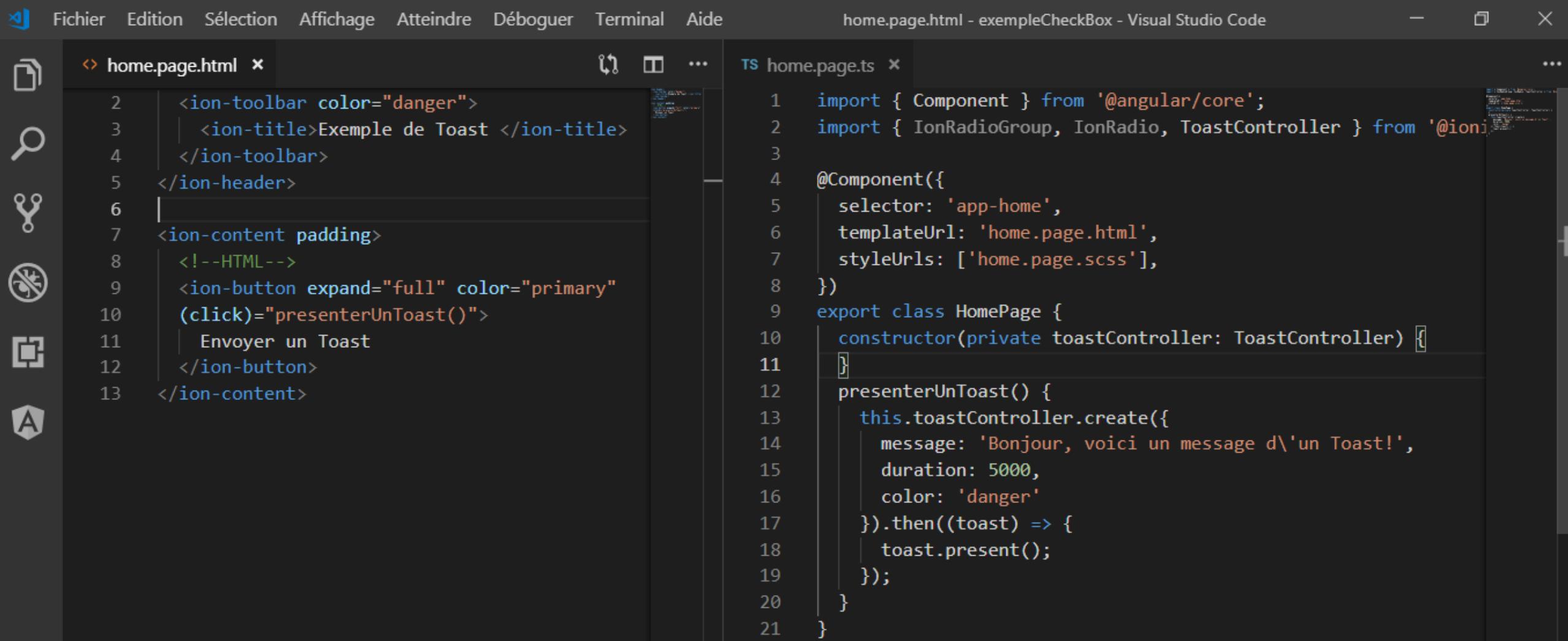
TS page-modale.page.ts ×

```
8  })
9  export class PageModalePage implements OnInit {
10    mesDonnees = {
11      nom: '',
12      fonction: ''
13    };
14    constructor(private modalController: ModalController,
15      private navParams: NavParams) { }
16
17    ngOnInit() {
18    }
19
20    ionViewWillEnter() {
21      this.mesDonnees = this.navParams.get('value');
22      alert('Données reçues: ' + this.mesDonnees.nom + ' ' + this.mesDonnees.fonction);
23    }
24
25    async fermerBoiteModale() {
26      this.mesDonnees.nom = 'Macky SALL, l\'ami des étudiants';
27      this.mesDonnees.fonction = 'Vainqueur des élections de 2019';
28      await this.modalController.dismiss(this.mesDonnees);
29    }
30
31  }
32 }
```

# Les Composants de l'Interface Utilisateur: Toast avec **ion-toast-controller** **ion-toast**



# Les Composants de l'Interface Utilisateur: Toast avec **ion-toast-controller** **ion-toast**



The screenshot shows the Visual Studio Code interface with two files open:

- home.page.html**: An Angular component template file containing HTML code for a toolbar, title, header, content, and a button that triggers a toast message.
- home.page.ts**: An Angular component type script file containing the component definition and a method to present a toast.

```
home.page.html
2 <ion-toolbar color="danger">
3   | <ion-title>Exemple de Toast </ion-title>
4 </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8   <!--HTML-->
9   <ion-button expand="full" color="primary"
10    (click)="presenterUnToast()">
11     Envoyer un Toast
12   </ion-button>
13 </ion-content>
```

```
home.page.ts
1 import { Component } from '@angular/core';
2 import { IonRadioGroup, IonRadio, ToastController } from '@ionic/angular';
3
4 @Component({
5   selector: 'app-home',
6   templateUrl: 'home.page.html',
7   styleUrls: ['home.page.scss'],
8 })
9 export class HomePage {
10   constructor(private toastController: ToastController) {}
11
12   presenterUnToast() {
13     this.toastController.create({
14       message: 'Bonjour, voici un message d\'un Toast!',
15       duration: 5000,
16       color: 'danger'
17     }).then((toast) => {
18       toast.present();
19     });
20   }
21 }
```

# Les Composants de l'Interface Utilisateur: **Popover** avec **ion-popover-controller**, **ion-popover**

- Un popover est une boîte de dialogue qui apparaît en haut de la page en cours.
- Un popover peut être créé en appelant la méthode **create**.
- Pour afficher un popover, appelez la méthode **present** sur une instance **PopoverController**.
- Pour fermer le popover après la création, appelez le **dismiss()** sur une instance **PopoverController**.

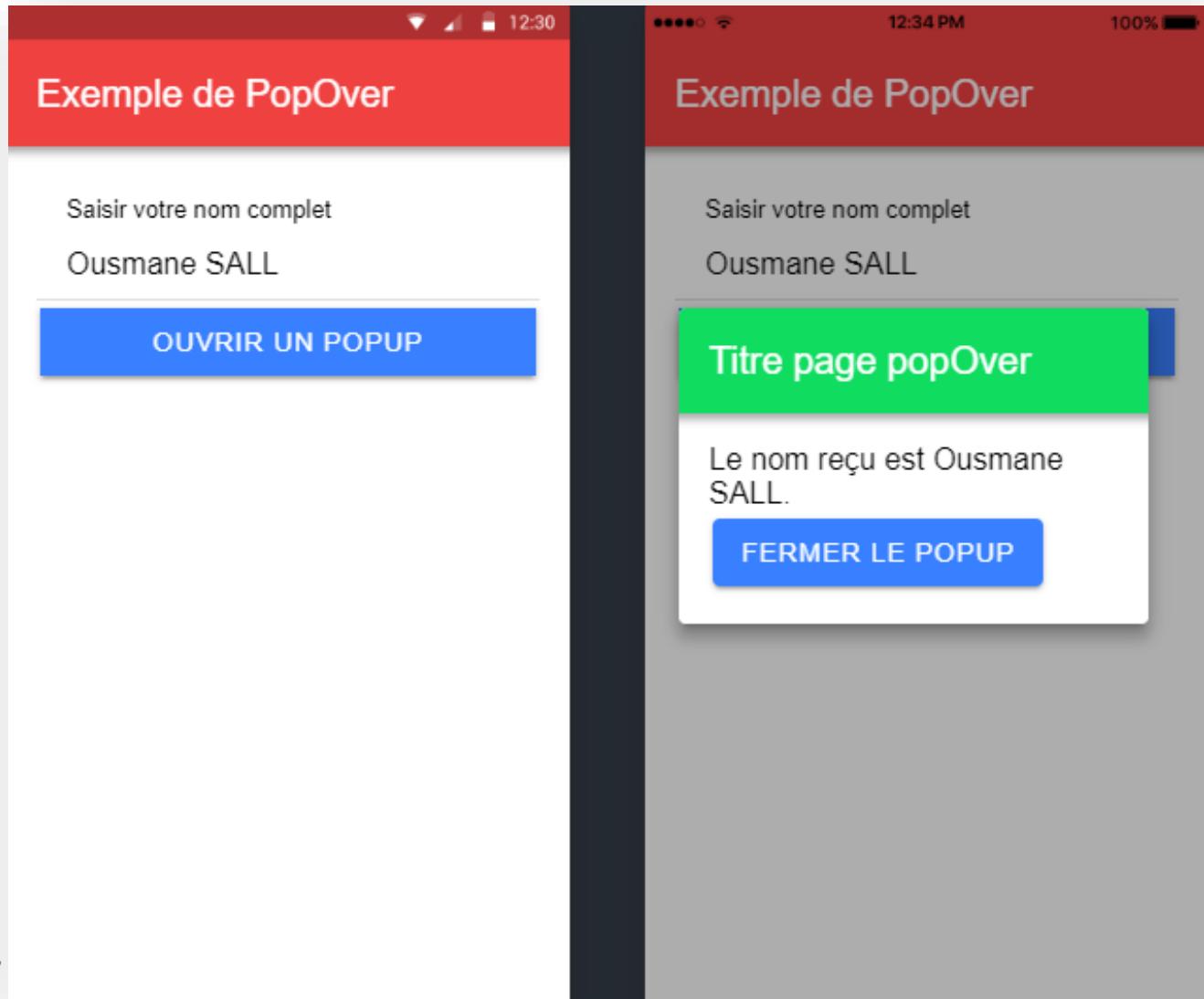
The screenshot shows the Ionic documentation for the `create(component, data, opts)` method. It includes a brief description: "Present a popover. See below for options". Below this is a table with three columns: Param, Type, and Details. The rows are:

Param	Type	Details
component	object	The Popover
data	object	Any data to pass to the Popover view
opts	PopoverOptions	Popover options

The screenshot shows the Ionic documentation for Popover Options. It has a table with three columns: Option, Type, and Description. The rows are:

Option	Type	Description
cssClass	string	Additional classes for custom styles, separated by spaces.
showBackdrop	boolean	Whether to show the backdrop. Default true.
enableBackdropDismiss	boolean	Whether the popover should be dismissed by tapping the backdrop. Default true.

# Les Composants de l'Interface Utilisateur: **Popover** avec **ion-popover-controller**, **ion-popover**

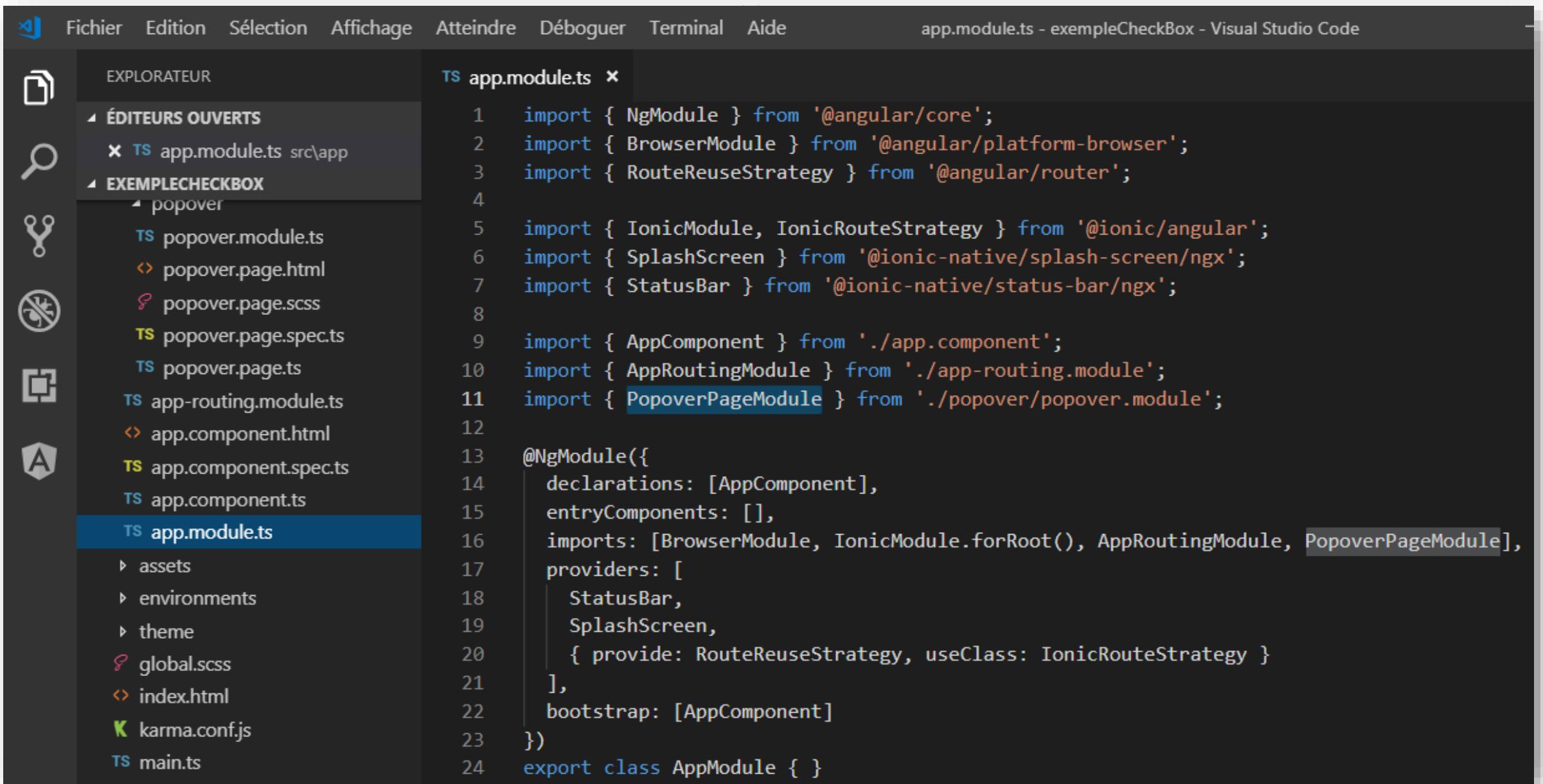


# Les Composants de l'Interface Utilisateur: **Popover** avec **ion-popover-controller**, **ion-popover**

- Générer une nouvelle page nommée popover

```
PS C:\Users\0Sall\Desktop\AppIonic\exempleCheckBox> ionic g page popover
> ng generate page popover
CREATE src/app/popover/popover.module.ts (548 bytes)
CREATE src/app/popover/popover.page.html (134 bytes)
CREATE src/app/popover/popover.page.spec.ts (698 bytes)
CREATE src/app/popover/popover.page.ts (260 bytes)
CREATE src/app/popover/popover.page.scss (0 bytes)
UPDATE src/app/app-routing.module.ts (455 bytes)
[OK] Generated page!
```

# Les Composants de l'Interface Utilisateur: Popover



The screenshot shows the Visual Studio Code interface with the following details:

- Menu Bar:** Fichier, Edition, Sélection, Affichage, Atteindre, Déboguer, Terminal, Aide.
- Title Bar:** app.module.ts - exempleCheckBox - Visual Studio Code
- Explorer View (Left):** Shows the project structure:
  - ÉDITEURS OUVERTS: app.module.ts (src\app)
  - EXEMPLECHECKBOX: popover (selected), popover.module.ts, popover.page.html, popover.page.scss, popover.page.spec.ts, popover.page.ts, app-routing.module.ts, app.component.html, app.component.spec.ts, app.component.ts, app.module.ts (highlighted).
  - assets, environments, theme, global.scss, index.html, karma.conf.js, main.ts.
- Code Editor (Right):** The content of app.module.ts is displayed:

```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { RouteReuseStrategy } from '@angular/router';
4
5 import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
6 import { SplashScreen } from '@ionic-native/splash-screen/ngx';
7 import { StatusBar } from '@ionic-native/status-bar/ngx';
8
9 import { AppComponent } from './app.component';
10 import { AppRoutingModule } from './app-routing.module';
11 import { PopoverPageModule } from './popover/popover.module';
12
13 @NgModule({
14   declarations: [AppComponent],
15   entryComponents: [],
16   imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule, PopoverPageModule],
17   providers: [
18     StatusBar,
19     SplashScreen,
20     { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
21   ],
22   bootstrap: [AppComponent]
23 })
24 export class AppModule { }
```

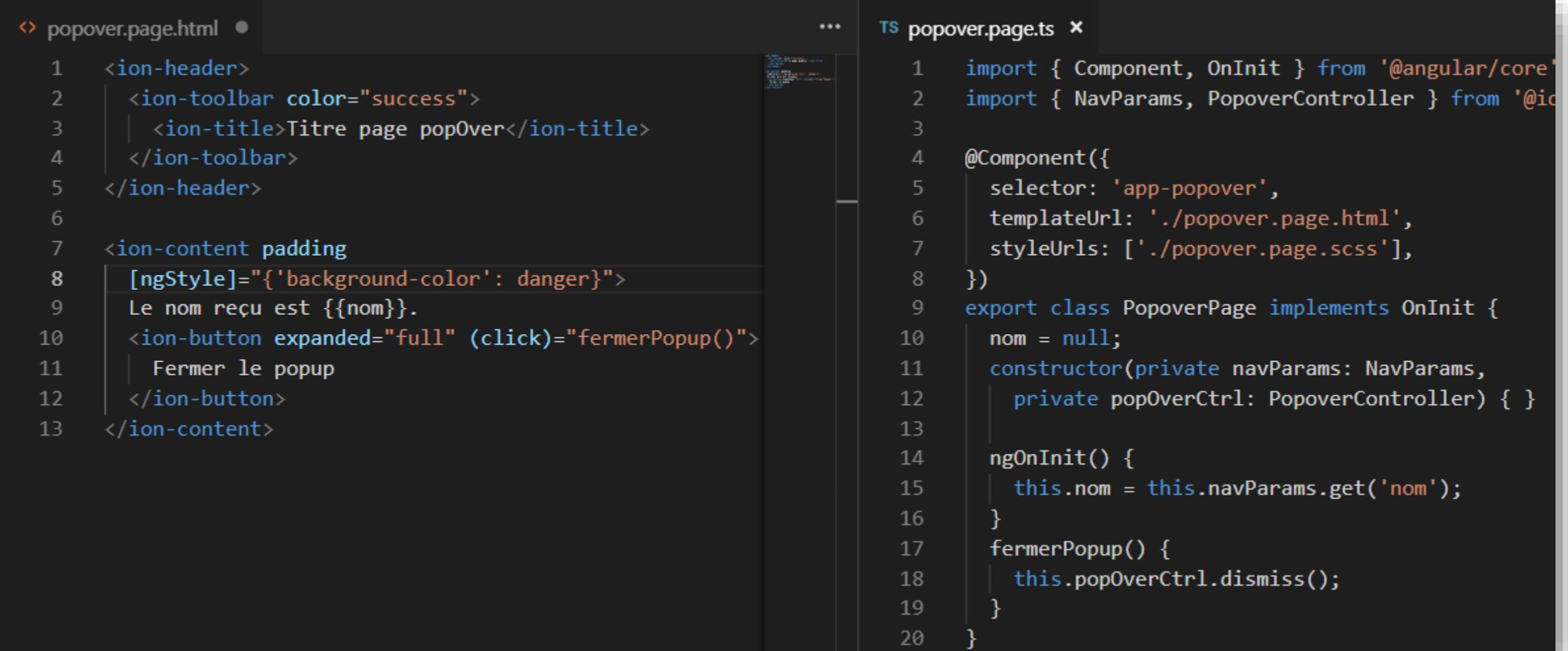
home.page.html

```
1 <ion-header>
2   <ion-toolbar color="danger">
3     <ion-title>Exemple de Toast </ion-title>
4   </ion-toolbar>
5 </ion-header>
6
7 <ion-content padding>
8
9 <ion-item>
10
11   <ion-label position="floating">
12     Saisir votre nom complet
13   </ion-label>
14
15   <ion-input placeholder="Nom et prénom"
16     [(ngModel)]="monNom">
17   </ion-input>
18
19 </ion-item>
20
21 <ion-button expand="full" color="primary"
22   (click)="presenterUnPopOver($event)">
23   Ouvrir un popup
24 </ion-button>
25
26 </ion-content>
```

TS home.page.ts

```
1 import { Component } from '@angular/core';
2 import { PopoverController } from '@ionic/angular';
3 import { PopoverPage } from '../popover popover.page';
4
5 @Component({
6   selector: 'app-home',
7   templateUrl: 'home.page.html',
8   styleUrls: ['home.page.scss'],
9 })
10 export class HomePage {
11   monNom = '';
12   constructor(private popOverController: PopoverController) {
13   }
14   async presenterUnPopOver(ev: Event) {
15     const popover = await this.popOverController.create({
16       component: PopoverPage,
17       event: ev,
18       componentProps: {
19         nom: this.monNom
20       },
21       translucent: true,
22     });
23     return await popover.present();
24   }
25 }
26
```

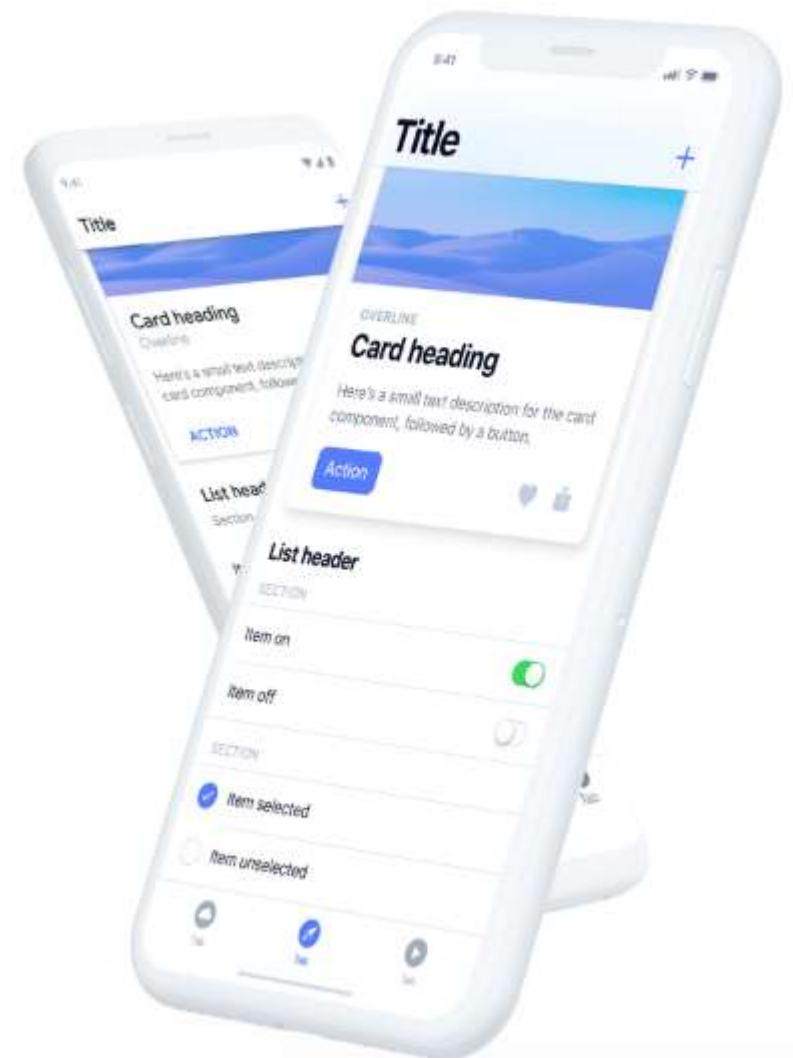
# Les Composants de l'Interface Utilisateur: **Popover** avec **ion-popover-controller**, **ion-popover**



The image shows a code editor with two files open:

- popover.page.html**: An Ionic template file containing an ion-header, ion-toolbar (color: success), ion-title (text: Titre page popOver), and an ion-content section with padding. Inside ion-content, there is a div with ngStyle="{'background-color': danger}" and text "Le nom reçu est {{nom}}". A button with expanded="full" and (click)="fermerPopup()" triggers a function to close the popover.
- popover.page.ts**: An Angular component file (class PopoverPage) implementing OnInit. It imports Component, OnInit, NavParams, and PopoverController from '@ionic/angular'. The component selector is 'app-popover', templateUrl is './popover.page.html', and styleUrls is ['./popover.page.scss']. The class has a constructor for NavParams and PopoverController. The ngOnInit() method sets the nom variable to the value from NavParams.get('nom'). The fermerPopup() method dismisses the popover using popOverCtrl.dismiss().

# Les Menus



# Les Composants de l'Interface Utilisateur: **FAB** **(Floating Action Button) List**

- Les fab sont des éléments de conteneur qui contiennent un ou plusieurs boutons de fab. Ils doivent être placés dans une position fixe qui ne défile pas avec le contenu. Fab devrait avoir un bouton principal.
- Les fabs peuvent également contenir des fab-lists contenant des boutons associés indiquant le clic sur le bouton principal. Un même conteneur de fabrication peut contenir plusieurs éléments de liste de fabrication avec des valeurs latérales différentes:
  - **ion-fab-button** <https://ionicframework.com/docs/api/fab-button>
  - **ion-fab-list** <https://ionicframework.com/docs/api/fab-list>
  - **ion-fab** <https://ionicframework.com/docs/api/fab>

# Les Composants de l'Interface Utilisateur: FAB List

- Propriétés:

<b>activated</b>	
Description	If <code>true</code> , both the <code>ion-fab-button</code> and all <code>ion-fab-list</code> inside <code>ion-fab</code> will become active. That means <code>ion-fab-button</code> will become a <code>close</code> icon and <code>ion-fab-list</code> will become visible.
Attribute	<code>activated</code>
<b>edge</b>	
Description	If <code>true</code> , the fab will display on the edge of the header if <code>vertical</code> is <code>"top"</code> , and on the edge of the footer if it is <code>"bottom"</code> . Should be used with a <code>fixed</code> slot.
Attribute	<code>edge</code>
Type	<code>boolean</code>

# Les Composants de l'Interface Utilisateur: FAB List

horizontal	
Description	Where to align the fab horizontally in the viewport.
Attribute	horizontal
Type	"center"   "end"   "start"   undefined
vertical	
Description	Where to align the fab vertically in the viewport.
Attribute	vertical
Type	"bottom"   "center"   "top"   undefined

# Les Composants de l'Interface Utilisateur: FAB List

## Methods

### close

#### Description

Close an active FAB list container

#### Signature

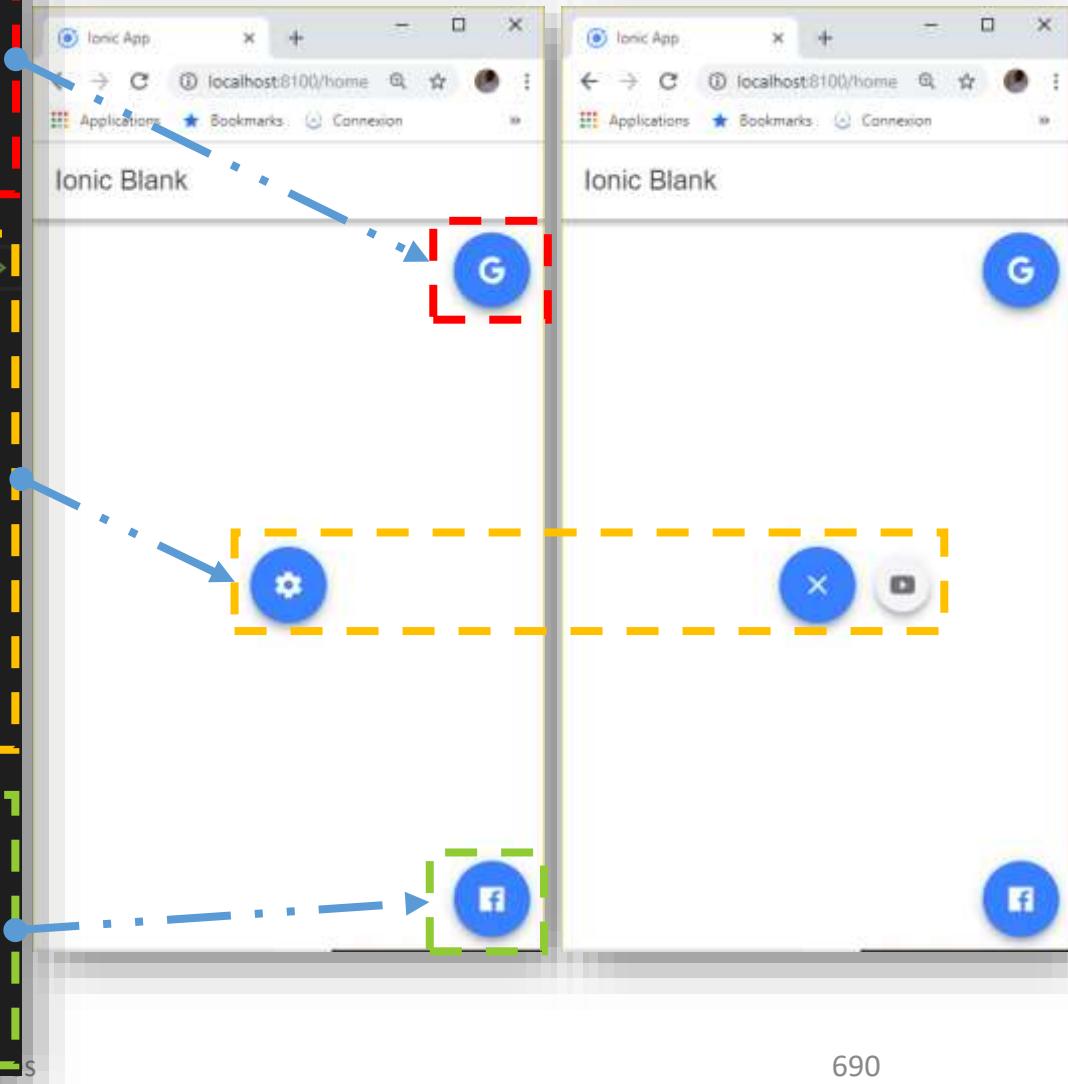
`close() => void`

# ce Utilisateur: FAB

```
<ion-content>
  <!-- Fab placé en haut à droite -->
  <ion-fab vertical="top" horizontal="end" slot="fixed">
    <ion-fab-button>
      <ion-icon name="logo-google"></ion-icon>
    </ion-fab-button>
  </ion-fab>

  <!-- fab placé au centre et contenant un autre fab visible au clic-->
  <ion-fab vertical="center" horizontal="center" edge slot="fixed">
    <ion-fab-button>
      <ion-icon name="settings"></ion-icon>
    </ion-fab-button>
    <ion-fab-list side="end">
      <ion-fab-button>
        <ion-icon name="logo-youtube"></ion-icon>
      </ion-fab-button>
    </ion-fab-list>
  </ion-fab>

  <!-- Fab placé en bas à droite -->
  <ion-fab vertical="bottom" horizontal="end" slot="fixed">
    <ion-fab-button>
      <ion-icon name="logo-facebook"></ion-icon>
    </ion-fab-button>
  </ion-fab>
```



# Les Composants de l'Interface Utilisateur: **Menu** **ion-menu-button**, **ion-menu-controller**, **ion-menu-toggle**, **ion-menu**, **ion-split-pane**

- **ion-menu-button** est un composant qui crée automatiquement l'icône et la fonctionnalité permettant d'ouvrir un menu sur une page. Voici ses propriétés: **autoHide**, **color**, **menu**, **mode**
- **ion-menu-controller** facilite le contrôle d'un menu. Les méthodes fournies peuvent être utilisées pour afficher le menu, activer le menu, basculer le menu, etc. Le contrôleur prendra une référence au menu à côté, ou id. si aucun de ces éléments ne lui est transmis, il prendra le premier menu trouvé. Méthodes:
  - close(menuId?: string | null | undefined) => Promise<boolean>
  - enable(shouldEnable: boolean, menuId?: string | null | undefined) => Promise<HTMLIonMenuItemElement | undefined>

# Les Composants de l'Interface Utilisateur: **Menu** **ion-menu-button**, **ion-menu-controller**, **ion-menu-toggle**, **ion-menu**, **ion-split-pane**

- **ion-menu-controller** Méthodes:

- `get(menuId?: string | null | undefined) => Promise<HTMLIonMenuItemElement | undefined>`
- `getMenus() => Promise<HTMLIonMenuItemElement[]>`
- `getOpen() => Promise<HTMLIonMenuItemElement | undefined>`
- `isAnimating() => Promise<boolean>`
- `isEnabled(menuId?: string | null | undefined) => Promise<boolean>`
- `isOpen(menuId?: string | null | undefined) => Promise<boolean>`
- `open(menuId?: string | null | undefined) => Promise<boolean>`
- `registerAnimation(name: string, animation: AnimationBuilder) => void`
- `swipeGesture(shouldEnable: boolean, menuId?: string | null | undefined) => Promise<HTMLIonMenuItemElement | undefined>`
- `toggle(menuId?: string | null | undefined) => Promise<boolean>`

# Les Composants de l'Interface Utilisateur: Menu

```
C:\Users\OSall\Desktop\AppIonic>ionic start exemplIonMenu --type=angular
```

```
Let's pick the perfect starter template!
```

```
Starter templates are ready-to-go Ionic apps that  
g you need to build your app. To bypass this  
prompt next time, supply template, the second arg.
```

```
? Starter template: blank
```

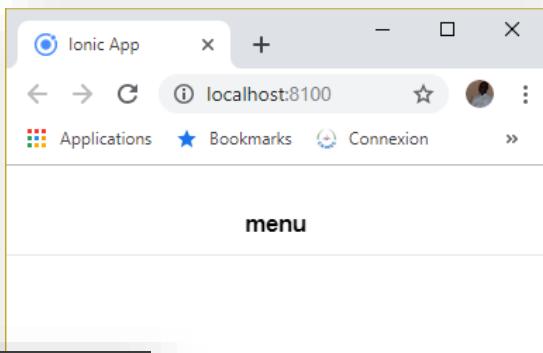
```
✓ Preparing directory .\exemplIonMenu - done!
```

```
(node:10584) ExperimentalWarning: The http2 modu:
```

```
✓ Downloading and extracting blank starter - done
```

```
PS C:\Users\OSall\Desktop\AppIonic\exemplIonMenu> ionic g page menu  
> ng generate page menu  
CREATE src/app/menu/menu.module.ts (533 bytes)  
CREATE src/app/menu/menu.page.html (131 bytes)  
CREATE src/app/menu/menu.page.spec.ts (677 bytes)  
CREATE src/app/menu/menu.page.ts (248 bytes)  
CREATE src/app/menu/menu.page.scss (0 bytes)  
UPDATE src/app/app-routing.module.ts (593 bytes)  
[OK] Generated page!  
PS C:\Users\OSall\Desktop\AppIonic\exemplIonMenu> ionic g page page1  
> ng generate page page1  
CREATE src/app/page1/page1.module.ts (538 bytes)  
CREATE src/app/page1/page1.page.html (132 bytes)  
CREATE src/app/page1/page1.page.spec.ts (684 bytes)  
CREATE src/app/page1/page1.page.ts (252 bytes)  
CREATE src/app/page1/page1.page.scss (0 bytes)  
UPDATE src/app/app-routing.module.ts (447 bytes)  
[OK] Generated page!  
PS C:\Users\OSall\Desktop\AppIonic\exemplIonMenu> ionic g page page2  
> ng generate page page2  
CREATE src/app/page2/page2.module.ts (538 bytes)  
CREATE src/app/page2/page2.page.html (132 bytes)  
CREATE src/app/page2/page2.page.spec.ts (684 bytes)  
CREATE src/app/page2/page2.page.ts (252 bytes)  
CREATE src/app/page2/page2.page.scss (0 bytes)  
UPDATE src/app/app-routing.module.ts (522 bytes)  
[OK] Generated page!
```

# Les Composants de l'Interface Utilisateur:



Fichier Edition Sélection Affichage Atteindre Déboguer Terminal Aide app-routing.module.ts - exemplonMenu - Visual Studio

EXPLORATEUR

EDITEURS OUVERTS

- x TS app-routing.module.ts src\app

EXEMPLONMENU

- e2e
- node\_modules
- src
  - app
    - home
    - menu
    - page1
    - page2

TS app-routing.module.ts \*

```
1 import { NgModule } from '@angular/core';
2 import { Routes, RouterModule } from '@angular/router';
3
4 const routes: Routes = [
5   { path: '', loadChildren: './menu/menu.module#MenuPageModule' },
6 ];
7
8 @NgModule({
9   imports: [RouterModule.forRoot(routes)],
10  exports: [RouterModule]
11 })
12 export class AppRoutingModule { }
```

# Les Composants de l'Interface Utilisateur: Menu

The screenshot shows the Visual Studio Code interface. The title bar reads "menu.module.ts - exemplonMenu - Visual Studio Code". The menu bar includes Fichier, Edition, Sélection, Affichage, Atteindre, Déboguer, Terminal, and Aide. The left sidebar has icons for Explorer, Search, Issues, and Preferences. The Explorer view shows a tree structure with "ÉDITEURS OUVERTS" containing "menu.module.ts" under "src\app\menu". It also shows "EXEMPLONMENU" with "src" expanded, containing "app" and "home". "app" contains "home.module.ts", "home.page.html", "home.page.scss", "home.page.spec.ts", and "home.page.ts". "menu" contains "menu.module.ts", "menu.page.html", "menu.page.scss", "menu.page.spec.ts", and "menu.page.ts". The "menu.module.ts" file is currently selected and shown in the main editor area.

```
TS menu.module.ts ×
1 import { NgModule } from '@ionic/angular';
2
3 import { MenuPage } from './menu.page';
4
5 const routes: Routes = [
6   {
7     path: 'menu',
8     component: MenuPage,
9     children: [
10       { path: 'page1', loadChildren: '../page1/page1.module#Page1PageModule' },
11       { path: 'page2', loadChildren: '../page2/page2.module#Page2PageModule' }
12     ]
13   },
14   {
15     path: '',
16     redirectTo: 'menu/page1'
17   }
18 ];
19
20 export const routing: Routes = [
21   ...routes
22 ];
```

# Les Composants de l'Interface Utilisateur: **Menu**

# Les Composants de l'Interface Utilisateur: **Menu**

# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles hybrides

## Plugins natives sous Ionic

<https://ionicframework.com/integrations>



# Les notifications

<https://github.com/katzer/cordova-plugin-local-notifications>

Une notification est un message que vous affichez à l'utilisateur en dehors de l'interface utilisateur normale de votre application. Lorsque vous indiquez au système d'émettre une notification, celle-ci apparaît d'abord sous forme d'icône dans la zone de notification. Pour voir les détails de la notification, l'utilisateur ouvre le tiroir de notification. La zone de notification et le tiroir de notification sont des zones contrôlées par le système que l'utilisateur peut afficher à tout moment.

## Properties

A notification does have a set of configurable properties. Not all of them are supported across all platforms.

Property	Property	Property	Property	Property	Property	Property	Property
id	data	timeoutAfter	summary	led	clock	channel	actions
text	icon	attachments	smallIcon	color	defaults	launch	groupSummary
title	silent	progressBar	sticky	vibrate	priority	mediaSession	foreground
sound	trigger	group	autoClear	lockscreen	number	badge	wakeup



# Les notifications

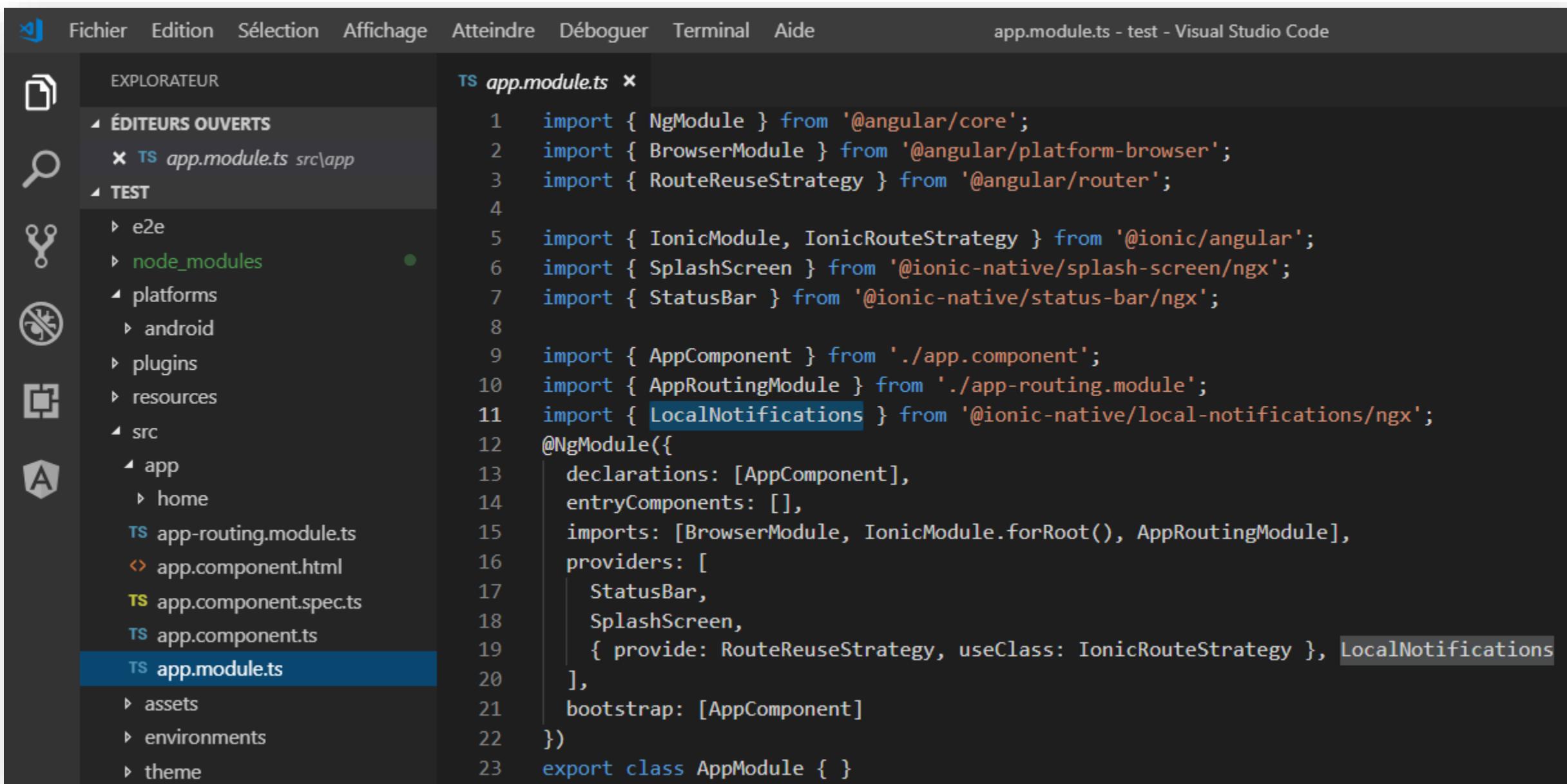
```
C:\Users\OSall\Desktop\AppIonic\test>npm install @ionic-native/local-notifications
npm WARN ajv-keywords@3.4.0 requires a peer of ajv@^6.9.1 but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {"os": "darwin", "arch": "any"} (current: {"os": "win32", "arch": "x64"})

+ @ionic-native/local-notifications@5.1.0
added 1 package from 1 contributor and audited 50845 packages in 45.227s
found 0 vulnerabilities

C:\Users\OSall\Desktop\AppIonic\test>ionic cordova plugin add cordova-plugin-local-notification
> ionic integrations enable cordova
[INFO] Downloading integration cordova
[INFO] Copying integrations files to project
CREATE resources
CREATE resources\splash.png
CREATE resources\ios
```

# Les notifications

<https://www.youtube.com/watch?v=1zKTLsJCrV0>

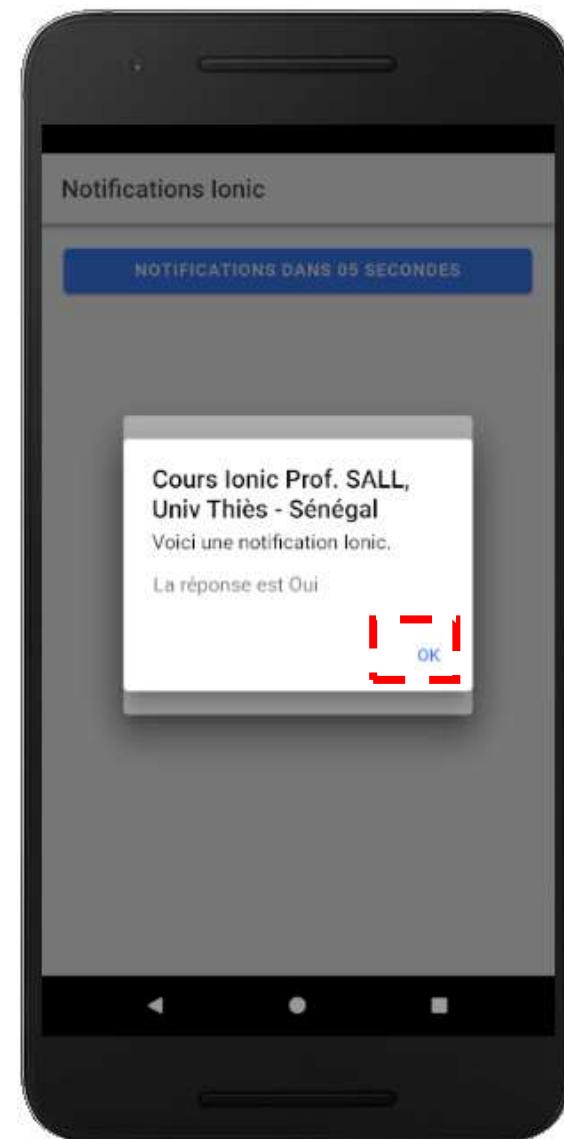
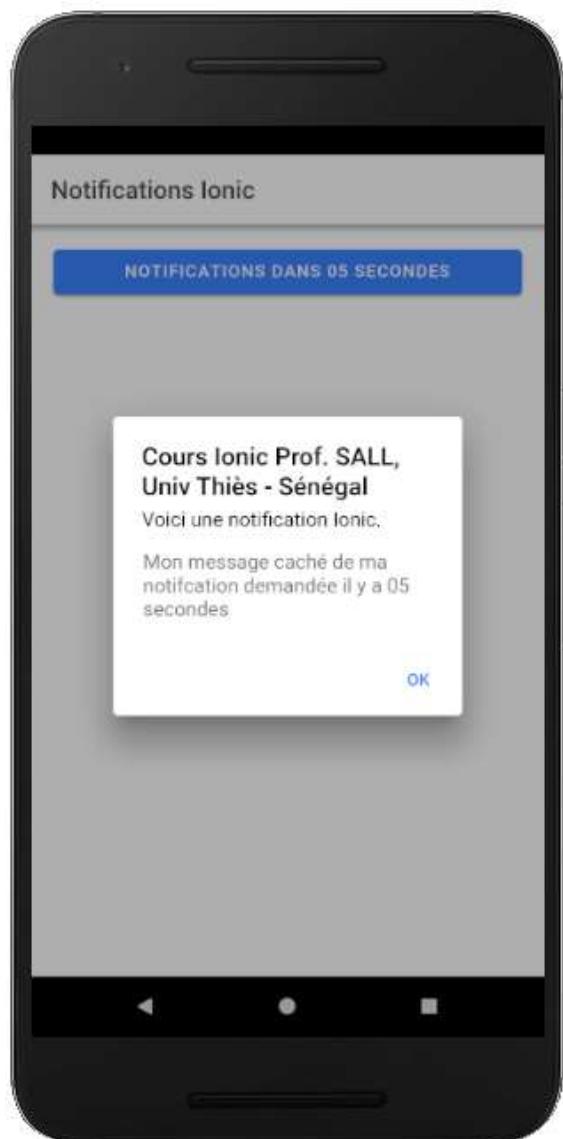
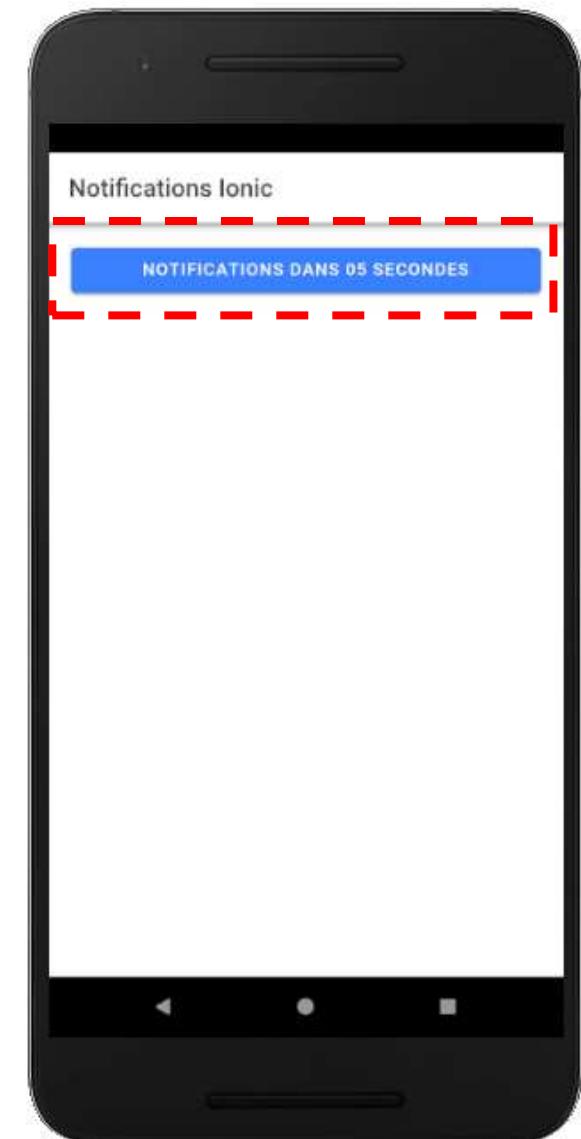


The screenshot shows the Visual Studio Code interface with the following details:

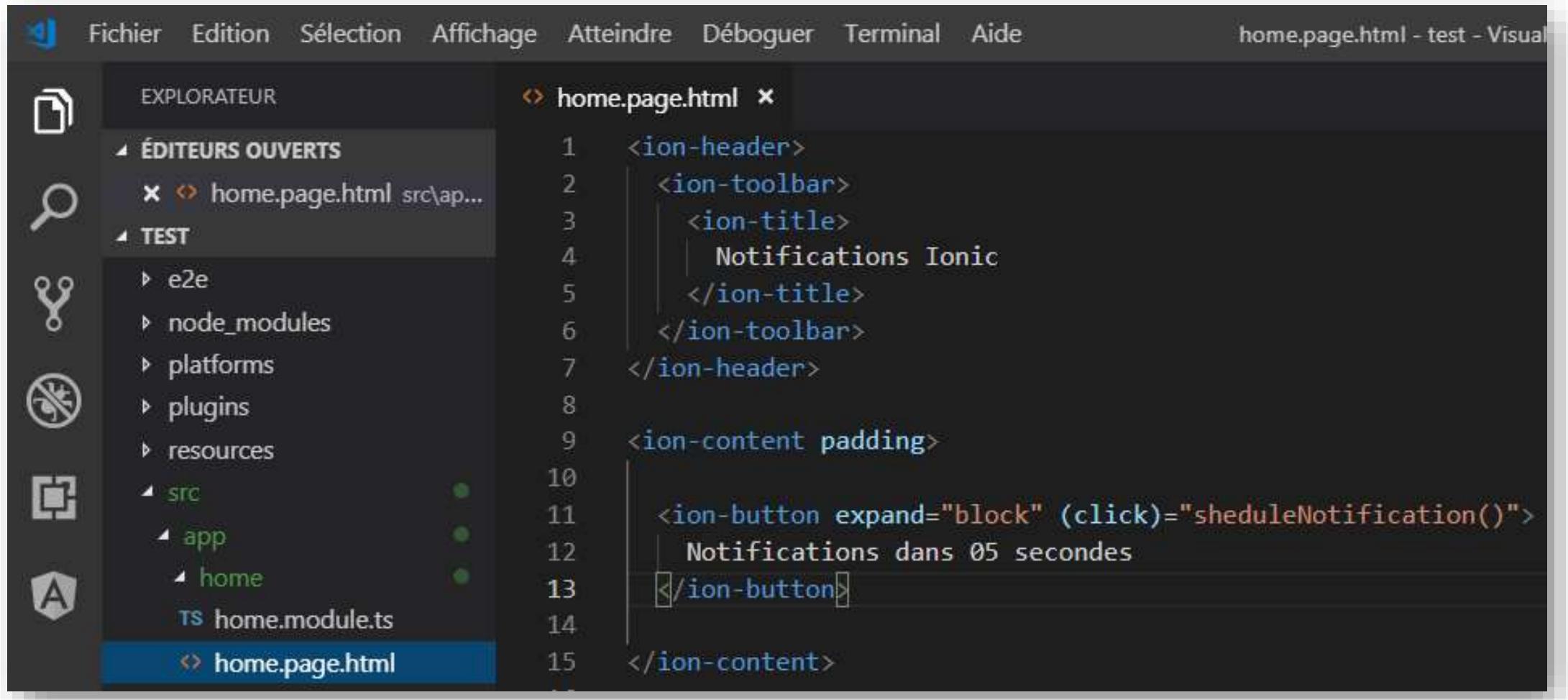
- Menu Bar:** Fichier, Edition, Sélection, Affichage, Atteindre, Déboguer, Terminal, Aide.
- Title Bar:** app.module.ts - test - Visual Studio Code
- Explorer View (Left):** Shows the project structure:
  - ÉDITEURS OUVERTS: app.module.ts (src\app)
  - TEST: e2e, node\_modules, platforms (android, plugins, resources), src (app, home), app-routing.module.ts, app.component.html, app.component.spec.ts, app.component.ts, app.module.ts (selected)
- Code Editor (Right):** The app.module.ts file content is displayed:

```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3 import { RouteReuseStrategy } from '@angular/router';
4
5 import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
6 import { SplashScreen } from '@ionic-native/splash-screen/ngx';
7 import { StatusBar } from '@ionic-native/status-bar/ngx';
8
9 import { AppComponent } from './app.component';
10 import { AppRoutingModule } from './app-routing.module';
11 import { LocalNotifications } from '@ionic-native/local-notifications/ngx';
12 @NgModule({
13   declarations: [AppComponent],
14   entryComponents: [],
15   imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule],
16   providers: [
17     StatusBar,
18     SplashScreen,
19     { provide: RouteReuseStrategy, useClass: IonicRouteStrategy },
20   ],
21   bootstrap: [AppComponent]
22 })
23 export class AppModule { }
```

# Les notifications: une notification individuelle



# Les notifications: une notification individuelle



The screenshot shows a dark-themed instance of Visual Studio Code. The top menu bar includes Fichier, Edition, Sélection, Affichage, Atteindre, Déboguer, Terminal, and Aide. The title bar indicates the file 'home.page.html - test - Visual Studio Code'. The left sidebar features icons for Explorer, Search, Test, and other development tools. The 'TEST' section is expanded, showing e2e, node\_modules, platforms, plugins, resources, and a 'src' folder which contains app and home subfolders, along with files home.module.ts and home.page.html. The 'home.page.html' file is currently selected and open in the main editor area. The code within the file is:

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Notifications Ionic
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10
11 <ion-button expand="block" (click)="scheduleNotification()">
12   Notifications dans 05 secondes
13 </ion-button>
14
15 </ion-content>
```

# Les notifications: une notification individuelle

```
constructor(private plt: Platform, private localNotifications: LocalNotifications,
private alertCtrl: AlertController) {

  this.plt.ready().then(() => {
    this.localNotifications.on('click').subscribe(res => {
      console.log('Click: ', res);
      let msg = res.data ? res.data.mydata : '';
      this.showAlert(res.title, res.text, msg);
    });
    this.localNotifications.on('trigger').subscribe(res => {
      console.log('Trigger: ', res);
      let msg = res.data ? res.data.mydata : '';
      this.showAlert(res.title, res.text, msg);
    });
  });
}
```

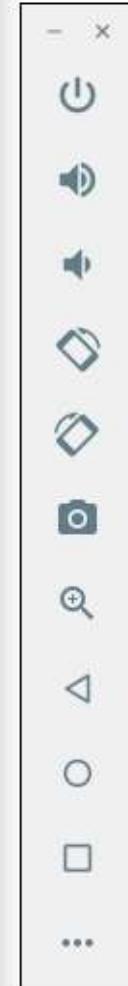
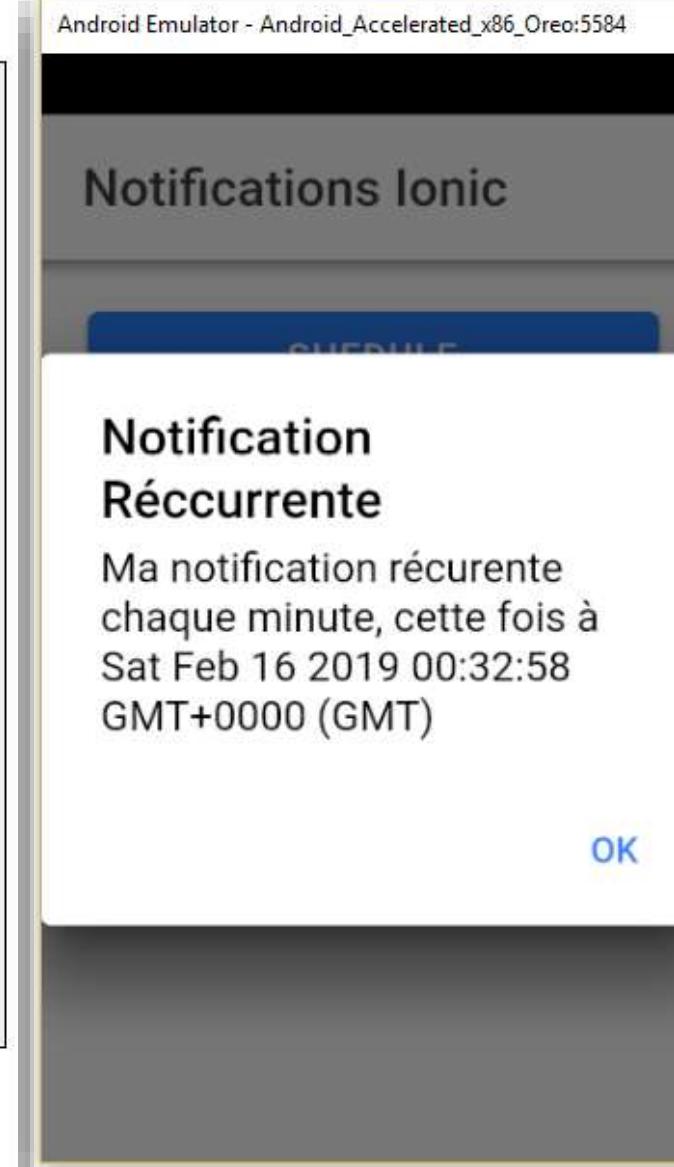
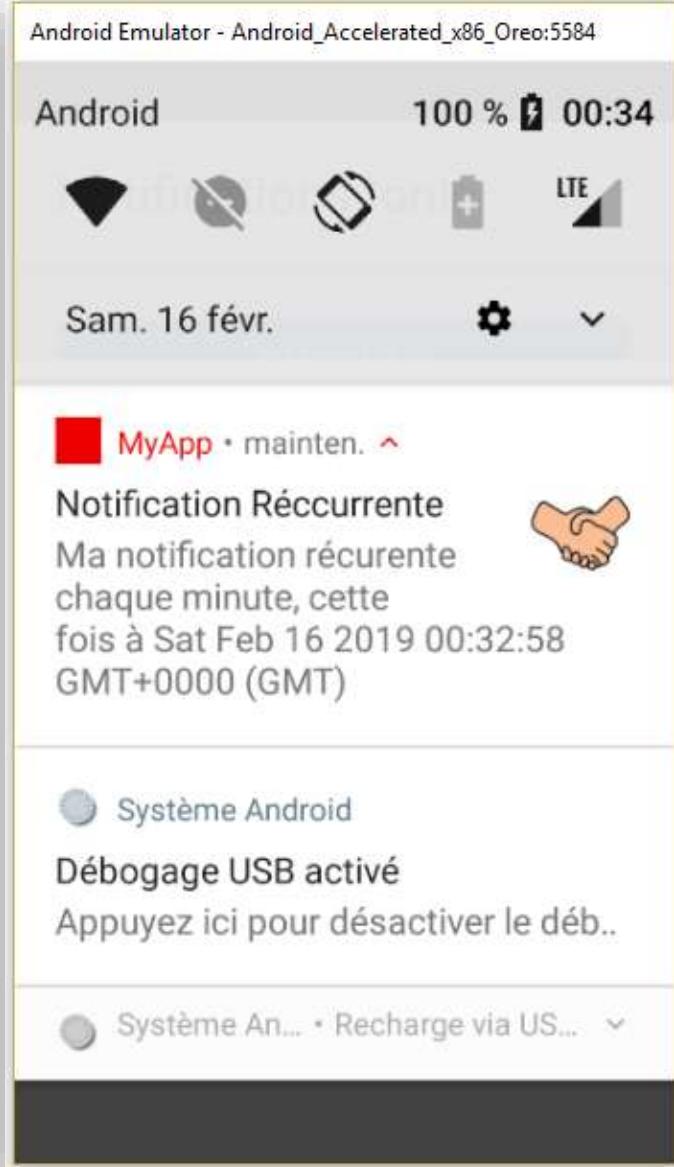
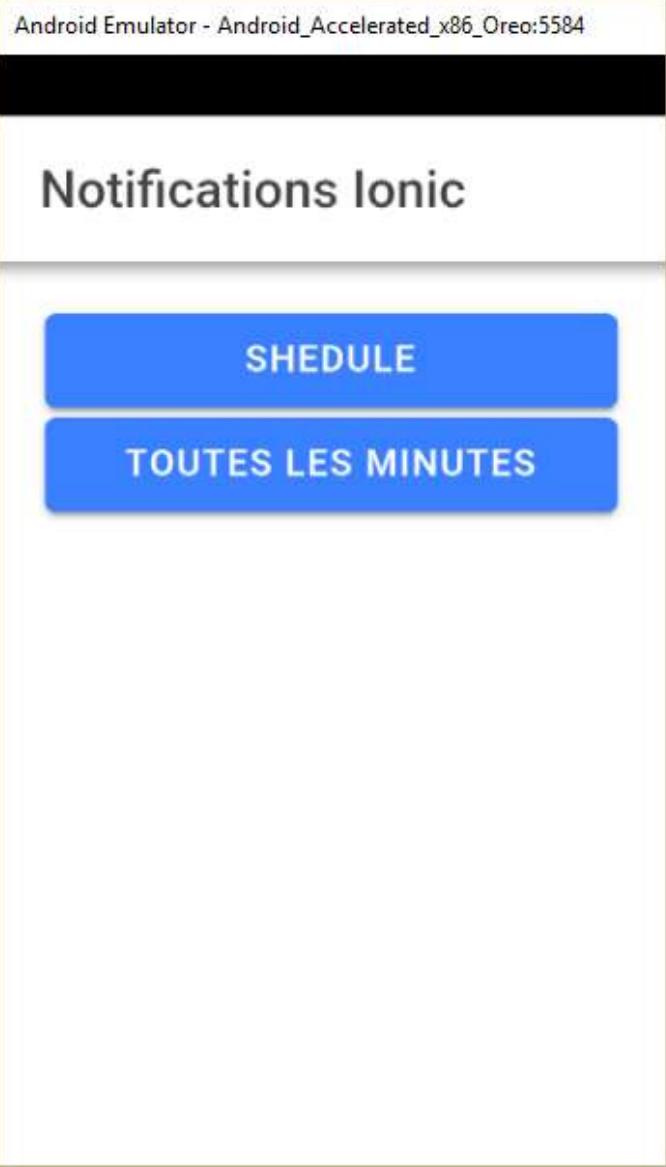


## TS home.page.ts ×

```
59  async scheduleNotification() {
60    const permission = await this.localNotifications.requestPermission();
61    if (permission) {
62      this.localNotifications.schedule({
63        id: 1,
64        title: 'Cours Ionic Prof. SALL, Univ Thiès - Sénégal',
65        text: 'Voici une notification Ionic.',
66        data: { mydata: 'Mon message caché de ma notification demandée il y a 05 secondes' },
67        trigger: { in: 5, unit: ELocalNotificationTriggerUnit.SECOND },
68        led: 'FF0FF0',
69        color: 'FF0000',
70        icon: 'file:///assets/ut.jpg',
71        vibrate: true,
72        sticky: false,
73        actions: [
74          { id: 'oui', title: 'Oui' },
75          { id: 'non', title: 'Non' }
76        ]
77      });
78
79      this.localNotifications.on('oui').subscribe(res => {
80        this.showAlert(res.title, res.text, 'La réponse est Oui');
81      });
82      this.localNotifications.on('non').subscribe(res => {
83        this.showAlert(res.title, res.text, 'La réponse est Non');
84      });
85    }
86  }
```

S

# Les notifications: une notification récurrente



# Les notifications: une notification récurrente

```
recurringNotification() {
  this.localNotifications.schedule({
    id: 22,
    title: 'Notification Récurrente',
    text: 'Ma notification récurrente chaque minute, cette fois à ' + (new Date()),
    trigger: { every: ELocalNotificationTriggerUnit.MINUTE },
    led: 'FF0FFF',
    color: 'FF0000',
    icon: 'https://img.icons8.com/doodle/2x/handshake.png',
  });
}
```

The screenshot shows a Visual Studio Code interface. The top bar includes the application logo, a search bar, and menu items: Fichier, Edition, Sélection, Affichage, Atteindre, Déboguer, Terminal, and Aide. The status bar indicates the file is "home.page.html - test - Visual Studio Code".

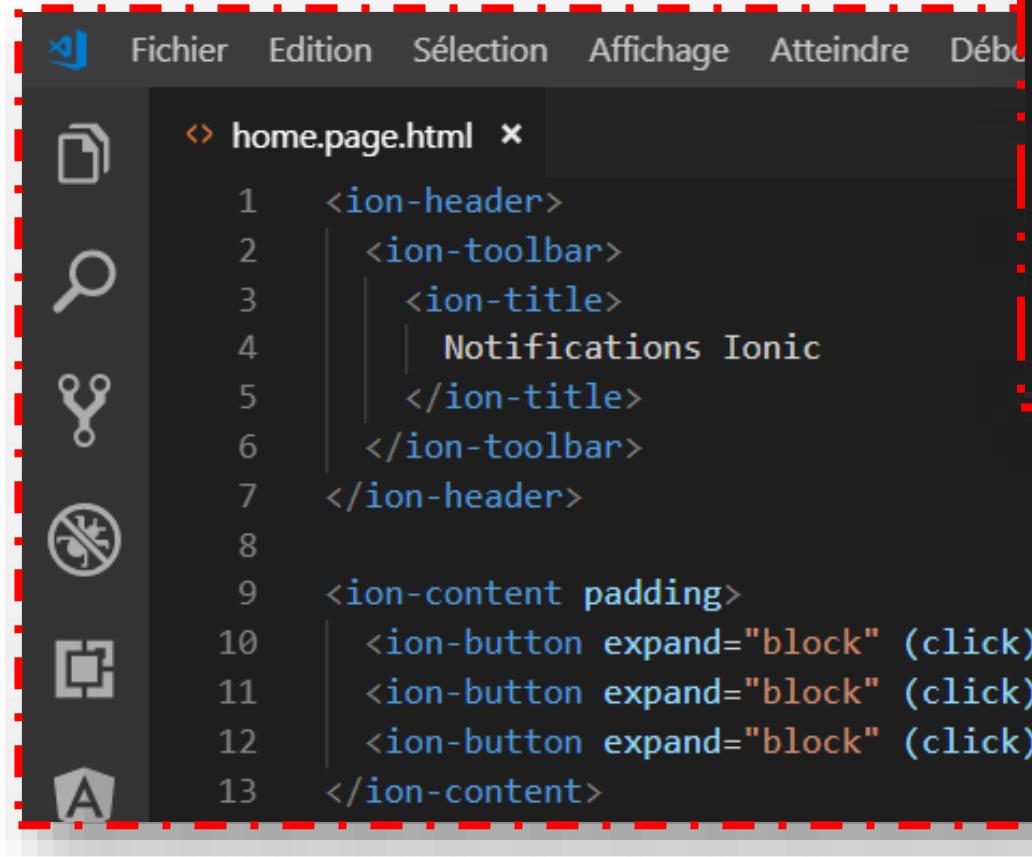
The main area displays the content of "home.page.html". The code uses the Ionic framework's HTML syntax:

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Notifications Ionic
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <ion-button expand="block" (click)="scheduleNotification()">Schedule</ion-button>
  <ion-button expand="block" (click)="recurringNotification()">Toutes les minutes</ion-button>
</ion-content>
```

The file explorer sidebar on the left lists files: "home.page.html" (marked with a checkmark), "index.html", "main.js", and "manifest.json".

# Les notifications: une notification à une date donnée



The screenshot shows a code editor with two files open. On the left, the file `home.page.html` contains the following Ionic code:

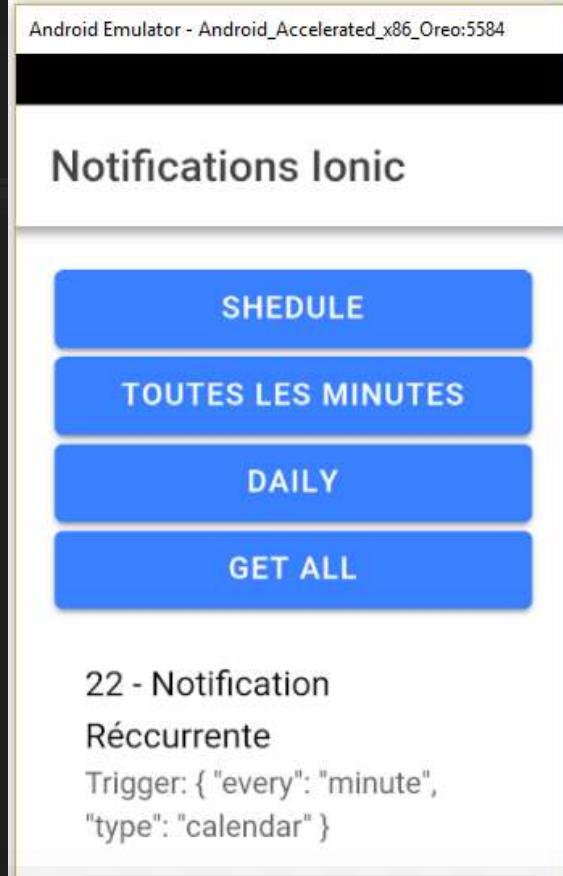
```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Notifications Ionic
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content padding>
10  <ion-button expand="block" (click)="sheduleNotification()">Schedule</ion-button>
11  <ion-button expand="block" (click)="recurringNotification()">Toutes les minutes</ion-button>
12  <ion-button expand="block" (click)="repeatingDailyNotification()">Daily</ion-button>
13 </ion-content>
```

```
repeatingDailyNotification() {
  this.localNotifications.schedule({
    id: 42,
    title: 'Bonjour',
    text: 'Ma notification à 22h30',
    trigger: { every: { hour: 22, minute: 30 } },
  });
}
```

# Les notifications: récupérer toutes les notifications

```
Fichier Edition Sélection Affichage Atteindre  
home.page.html x  
1 <ion-header>  
2   <ion-toolbar>  
3     <ion-title>  
4       Notifications Ionic  
5     </ion-title>  
6   </ion-toolbar>  
7 </ion-header>  
8  
9 <ion-content padding>  
10  <ion-button expand="block" (click)="sheduleNotification()">Shedule</ion-button>  
11  <ion-button expand="block" (click)="recurringNotification()">Toutes les minutes</ion-button>  
12  <ion-button expand="block" (click)="repeatingDailyNotification()">Daily</ion-button>  
13  <ion-button expand="block" (click)="getAllNotification()">Get All</ion-button>  
14 <ion-list>  
15   <ion-item *ngFor="let n of scheduled">  
16     <ion-label text-wrap>  
17       {{n.id}} - {{n.title}}  
18       <p>Trigger: {{n.trigger | json}}</p>  
19     </ion-label>  
20   </ion-item>  
21 </ion-list>  
22 </ion-content>
```

```
getAllNotification() {  
  this.localNotifications.getAll().then(res => {  
    this.scheduled = res;  
  });  
}
```



```
cancelAllNotification() {  
  this.localNotifications.cancelAll();  
}
```



TypeScript



# INF 3511 Programmation des Mobiles: Développement d'Applications Mobiles Natives



Licence Informatique Option Génie Logiciel

Année Universitaire 2018-2019

Ousmane SALL

Maître de Conférences CAMES

Université de THIES - UFR Sciences et Technologies -Département Informatique

Partie 6:

Développement d'Applications Natives avec Android



# A propos de moi

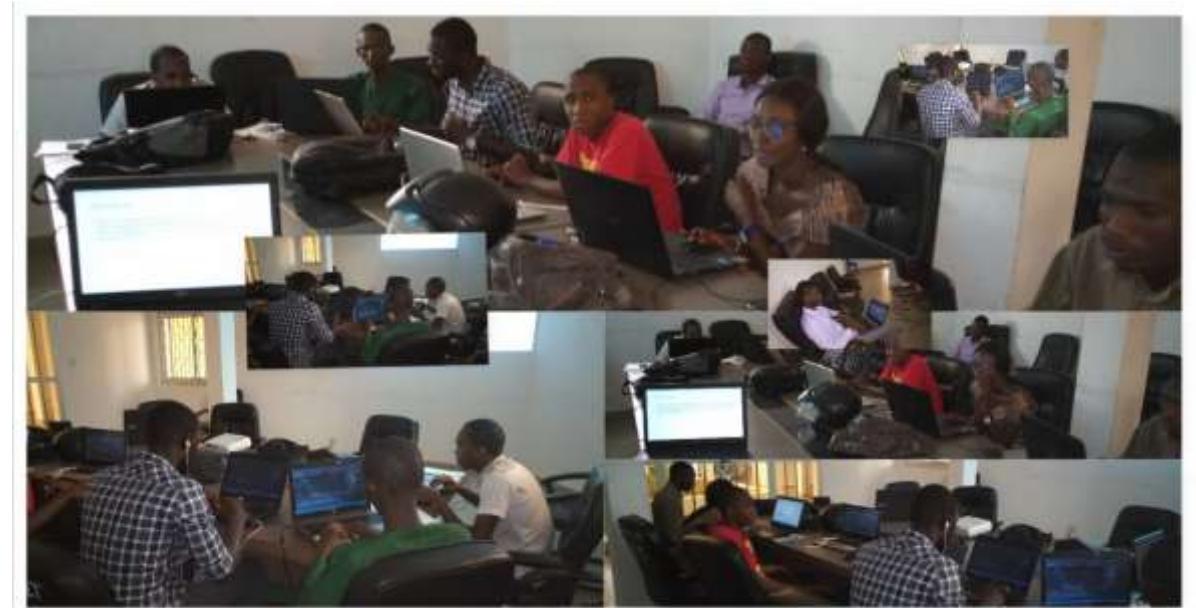


- Enseignant-Chercheur à l'UFR SET- Université de THIES <http://sites.univ-thies.sn/osall751/>
- Enseignements:
  - Algorithmique et Programmation(C, Java, PHP)
  - Programmation WEB dynamique(HTML 5 CSS, PHP, MySQL, CMS,...)
  - Programmation Java
  - Programmation JavaEE, JSF, Spring
  - Technologies Mobiles Android, Xamarin, Ionic
  - Programmation .Net, C#
  - Gestion de Projet Informatique
  - Génie Logiciel
- Contact:
  - [osall@univ-thies.sn](mailto:osall@univ-thies.sn)
  - UFR SET, Université de THIES -Dpt Informatique, BP 967 THIES.



# Une sagesse chinoise...

*« J'écoute et j'oublie; je lis et je comprends; je fais et j'apprends »*  
[Proverbe chinois]



# Contenu

1. Généralités sur les Technologies Mobiles
2. Périphériques et Systèmes d'exploitations mobiles
3. Approches de développement mobile
4. Xamarin pour le développement d'Applications Mobiles
5. Développement d'Applications Mobiles Hybrides avec Cordova et le framework Ionic 4
6. **Développement d'Applications Natives avec Android**



# Rappel



## Android

Système d'exploitation

Android, prononcé à la française /ɑ̃.dʁɔ.i/, en anglais /æn.droɪd/, est un système d'exploitation mobile fondé sur le noyau Linux et développé actuellement par Google. [Wikipédia](#)

Date de sortie initiale : 23 septembre 2008

Dernière version stable : 9.0 (6 août 2018)

Entreprise / Fondateur : Andy Rubin et Rich Miner (en)

Type de noyau : Monolithique, (noyau Linux modifié)

Entreprise / Développeur : Open Handset Alliance et Google

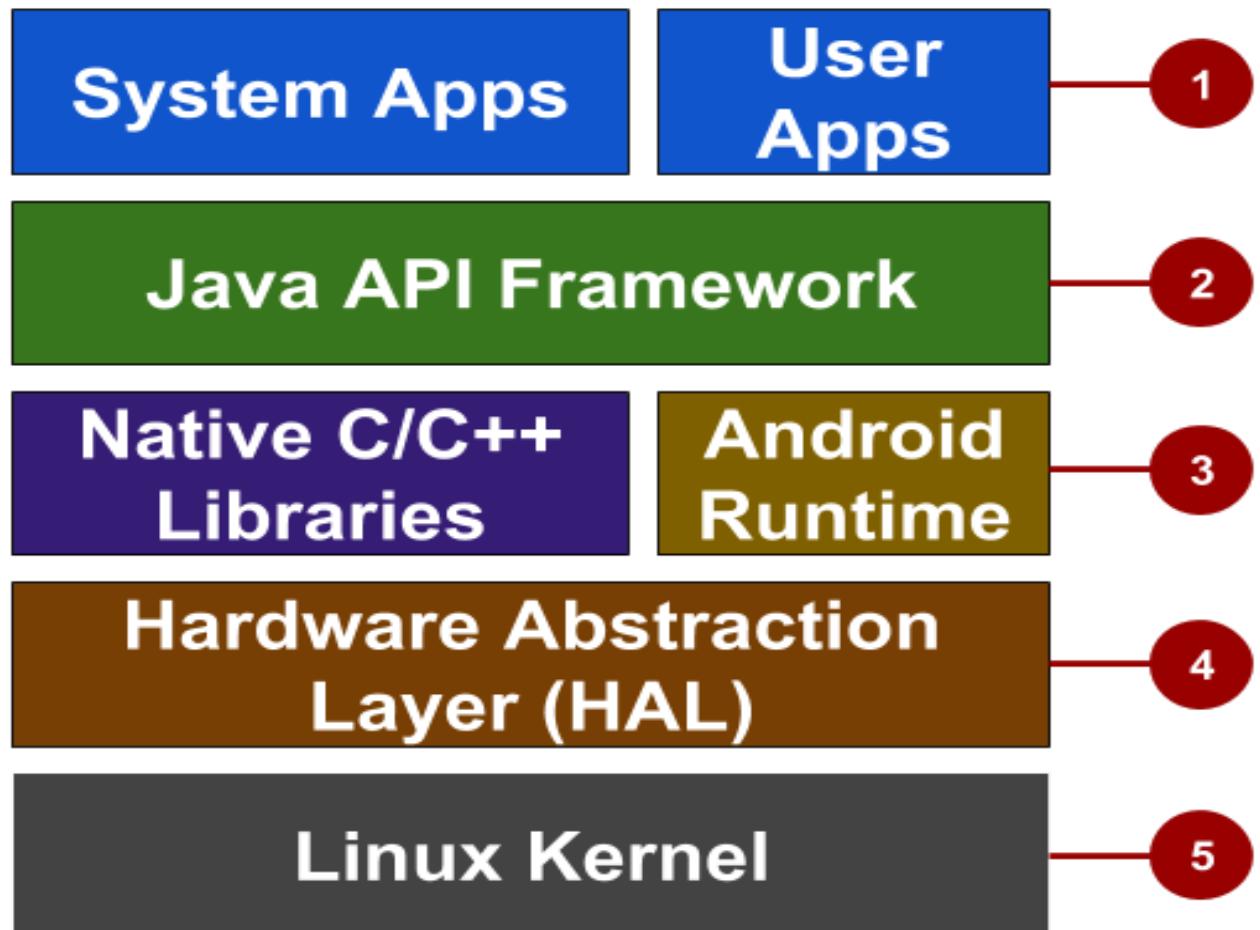
Programmé en : Java, C, C++, Extensible Markup Language, Assembleur, Python, Script shell, Go, make, D

Android	
	Current logomark (top) and logotype (bottom)
<a href="#">Screenshot</a>	[show]
Developer	Google, Open Handset Alliance
Written in	Java (UI), C (core), C++ and others <sup>[1]</sup>
OS family	Unix-like (Modified Linux kernel)
Working state	Current
Source model	Open source (most devices include proprietary components, such as Google Play)
Initial release	September 23, 2008; 10 years ago <sup>[2]</sup>
Latest release	9.0 "Pie" / August 6, 2018; 7 months ago
Latest preview	Android Q Beta 1 <sup>[3]</sup> / March 13, 2019; 21 days ago
Marketing target	Smartphones, tablet computers, smartTVs (Android TV), Android Auto and smartwatches (Wear OS)
Available in	100+ languages <sup>[4]</sup>
Update method	Over-the-air
Package manager	APK (primarily through Google Play; installation of APKs also possible locally or from alternative sources such as F-Droid)
Platforms	32- and 64-bit ARM, x86 and x86-64
Kernel type	Monolithic
Userland	Bionic libc, <sup>[5]</sup> mksh shell, <sup>[6]</sup> Toybox as core utilities (beginning with Android 6.0) <sup>[7][8]</sup>
Default user interface	Graphical (multi-touch)
License	Apache License 2.0 GNU GPL v2 for the Linux kernel modifications <sup>[9]</sup>
Official website	<a href="http://www.android.com">www.android.com</a> <sup>[9]</sup>
Articles in the series	
<a href="#">Android version history</a>	

# ANDROID PLATFORM ARCHITECTURE

# Android stack

1. System and user apps
2. Android OS API in Java framework
3. Expose native APIs; run apps
4. Expose device hardware capabilities
5. Linux Kernel



# Older Android versions



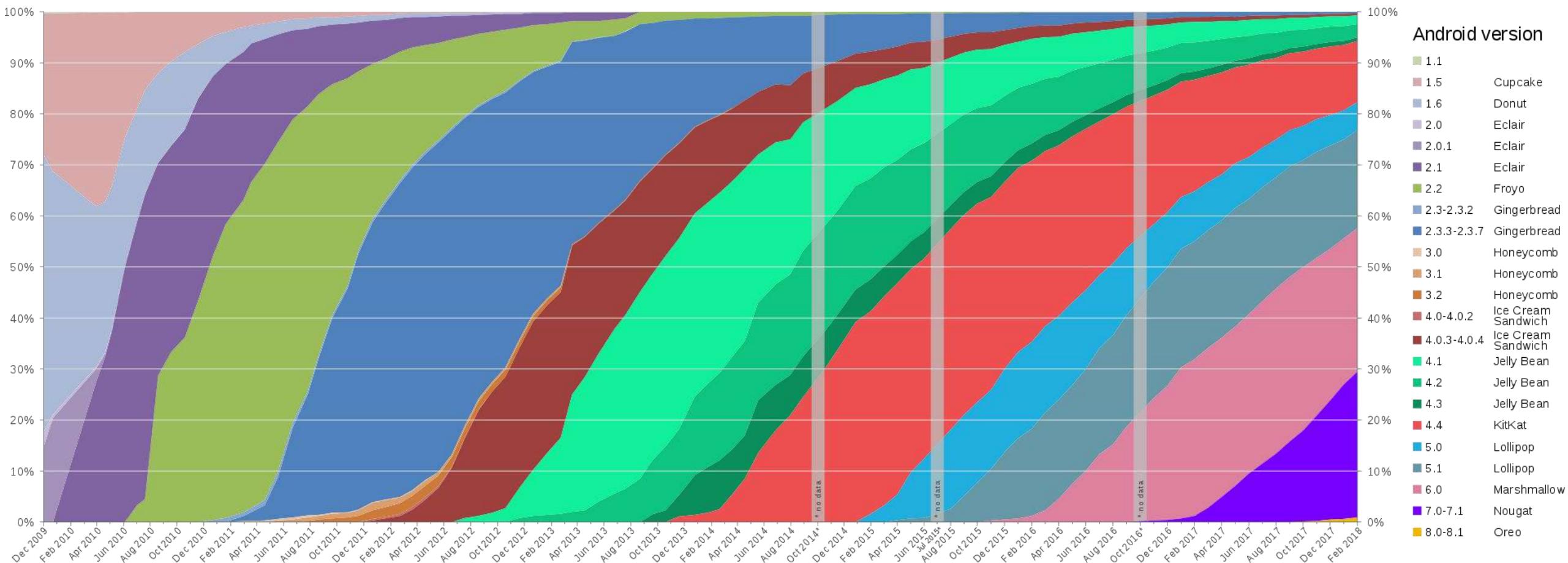
Codename	Version	Released	API Level
<b><i>Honeycomb</i></b>	3.0 - 3.2.6	Feb 2011	11 - 13
<b><i>Ice Cream Sandwich</i></b>	4.0 - 4.0.4	Oct 2011	14 - 15
<b><i>Jelly Bean</i></b>	4.1 - 4.3.1	July 2012	16 - 18
<b><i>KitKat</i></b>	4.4 - 4.4.4	Oct 2013	19 - 20
<b><i>Lollipop</i></b>	5.0 - 5.1.1	Nov 2014	21 - 22



# Newer Android versions

Codename	Version	Released	API Level
<b><i>Marshmallow</i></b>	6.0 - 6.0.1	Oct 2015	23
<b><i>Nougat</i></b>	7.0 - 7.1	Sept 2016	24 - 25
<b><i>Oreo</i></b>	8.0 - 8.1	Sept 2017	26 - 27
<b><i>Pie</i></b>	9.0	Aug 2018	28

# Rappel



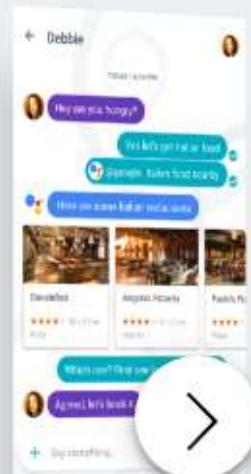
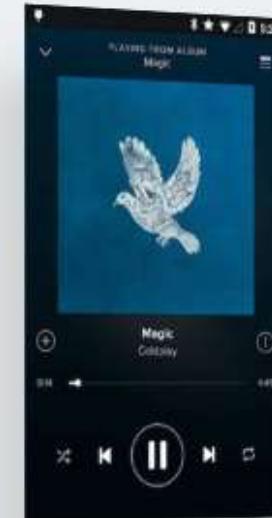
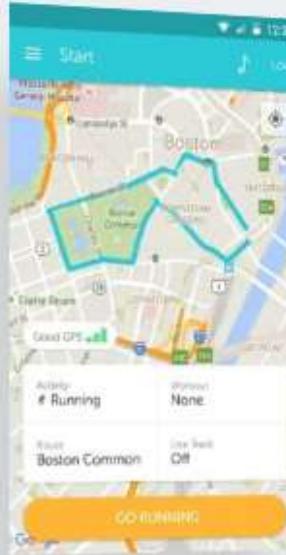
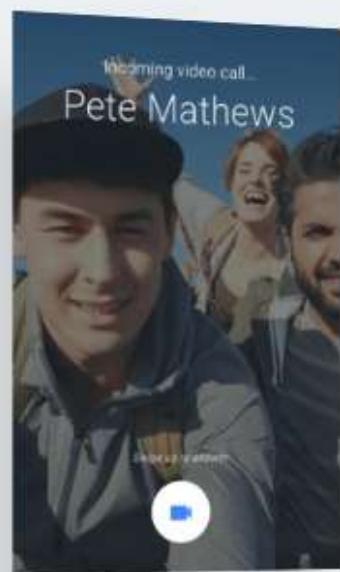
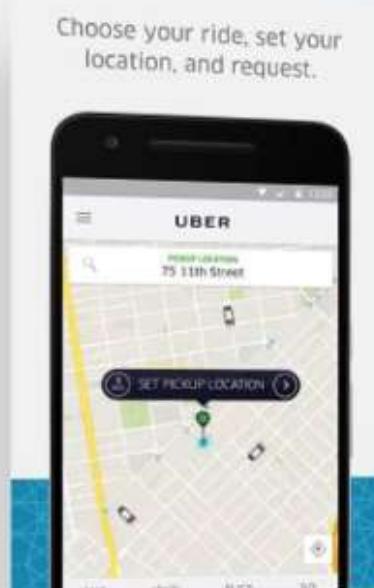
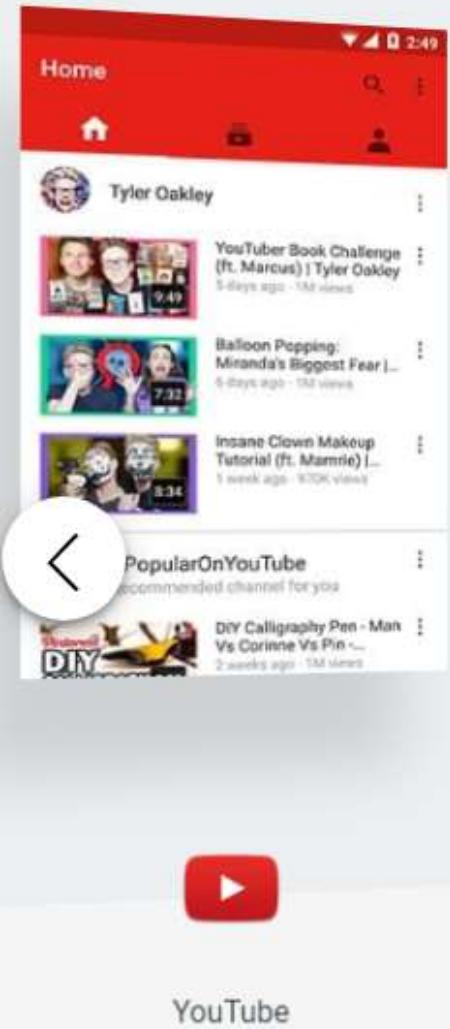
[https://fr.wikipedia.org/wiki/Historique\\_des\\_versions\\_d%27Android](https://fr.wikipedia.org/wiki/Historique_des_versions_d%27Android)

# Introducing Android 9 Pie

Powered by AI to make your smartphone  
smarter, faster and tailored to you.

[LEARN MORE](#)





YouTube

Uber



Duo



Runkeeper



Spotify

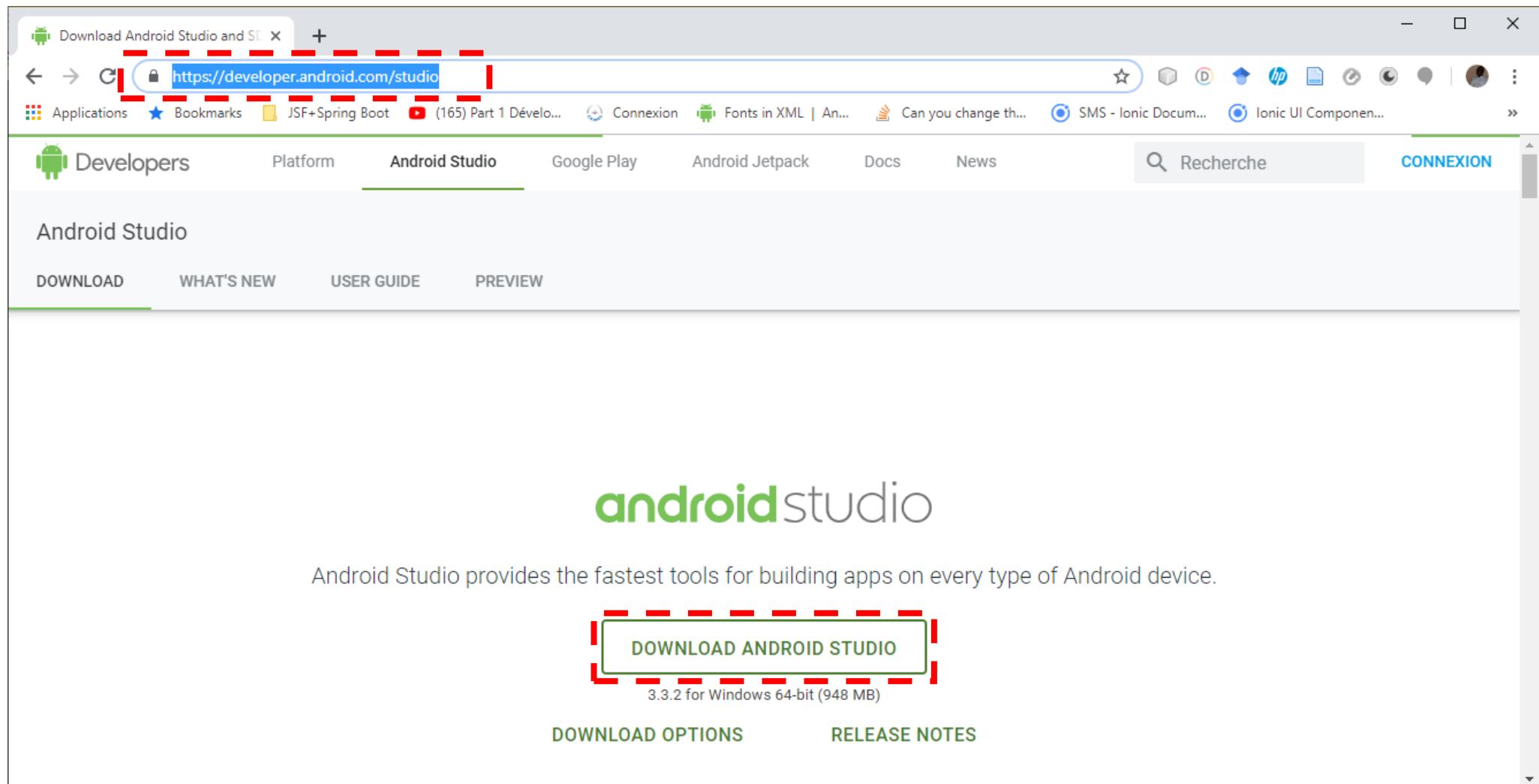


Allo

# INF 3511 Programmation des Mobiles: Développement d'Applications Natives avec Android

**PRÉPARATION DE L'ENVIRONNEMENT DE  
DÉVELOPPEMENT ET CONCEPTS DE BASE**



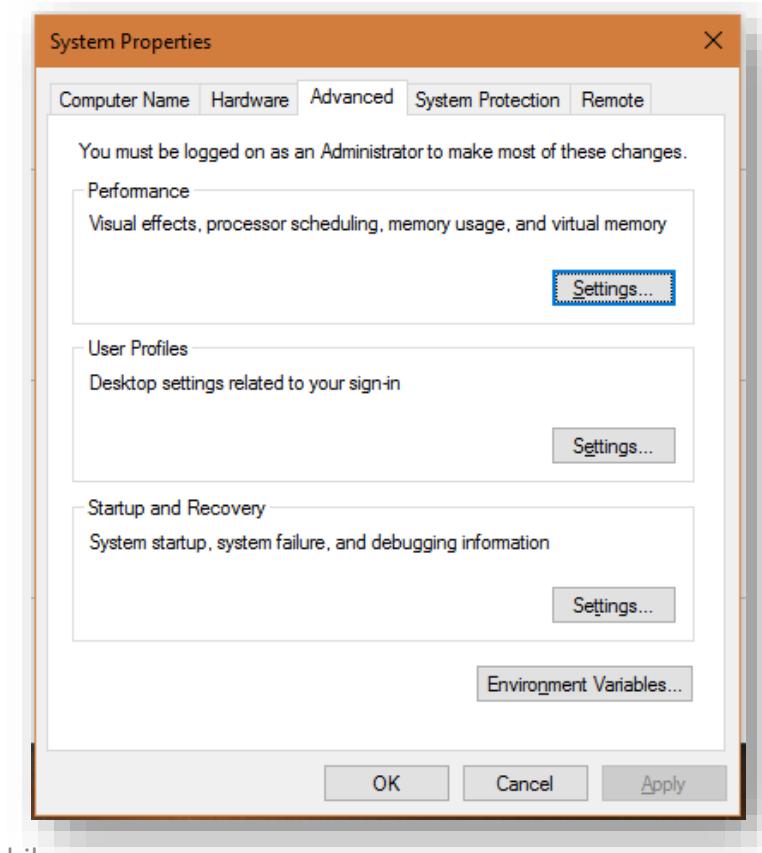


# Preparation de l'ordinateur

## Java : JDK (Java Developper Kit)

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

- Installer Java JDK  
<http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk>
- Ajouter une variable system
  - Variables d' Environment
    - Nouveau
      - Nom de la Variable : JDK\_HOME
      - Valeur de la variable : Path au repertoire Java (C:\Program Files (x86)\Java)
  - Installation Android Studio  
<https://developer.android.com/studio/index.html>

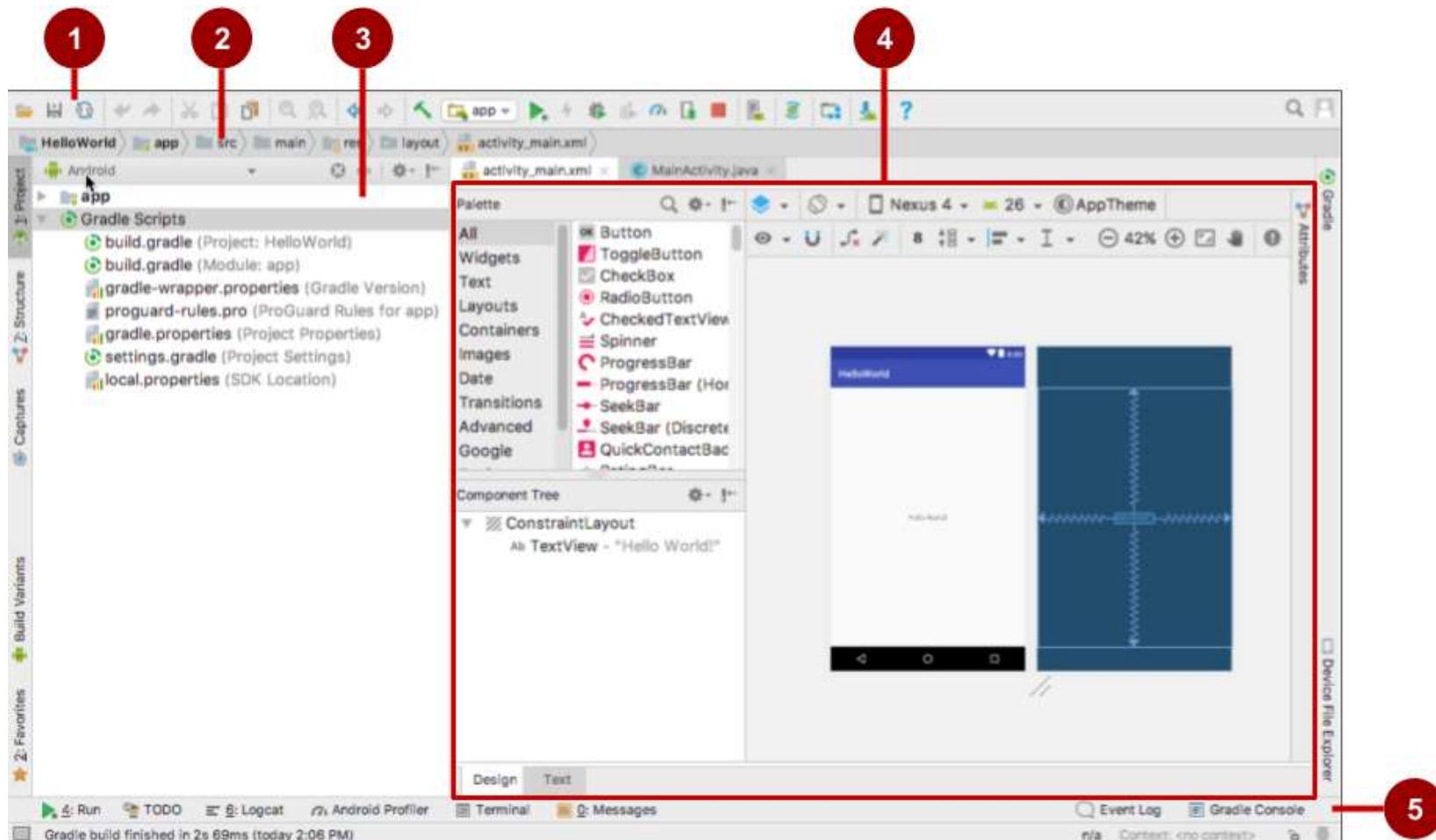


# Android Studio

# What is Android Studio?

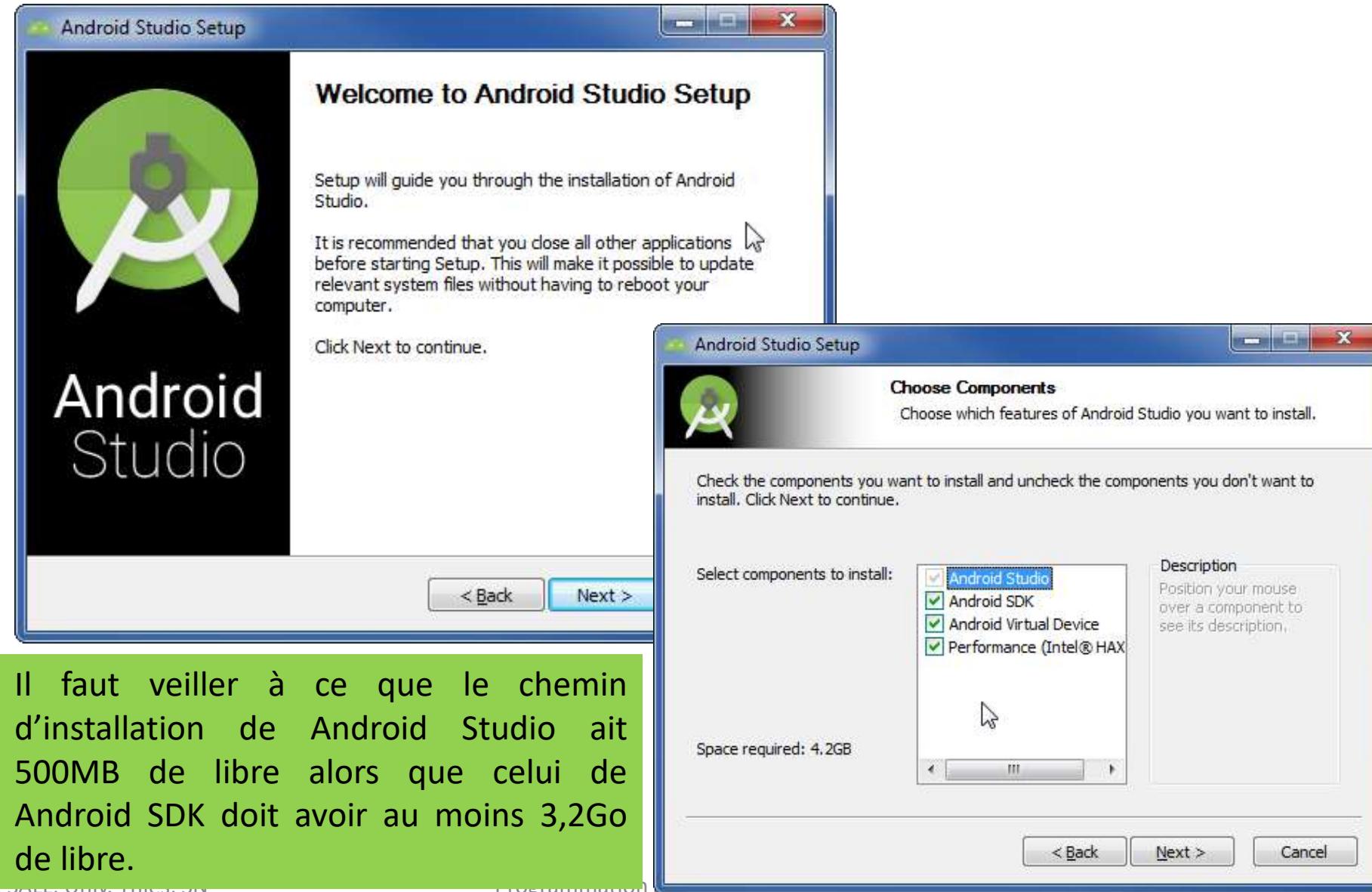
- Android integrated development environment (IDE)
- Project and Activity templates
- Layout editor
- Testing tools
- Gradle-based build
- Log console and debugger
- Emulators

# Android Studio interface

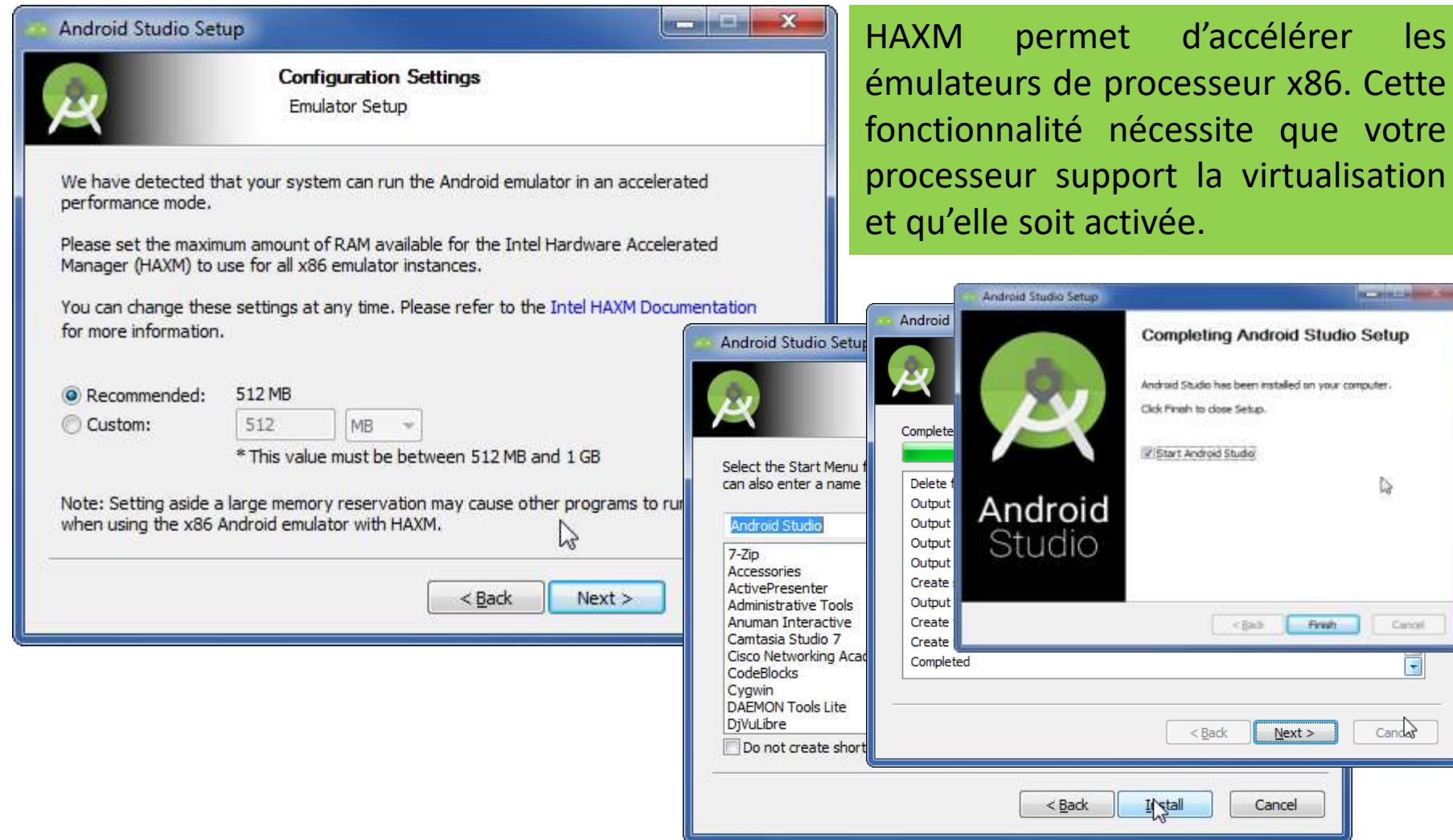


1. Toolbar
2. Navigation bar
3. Project pane
4. Editor
5. Tabs for other panes

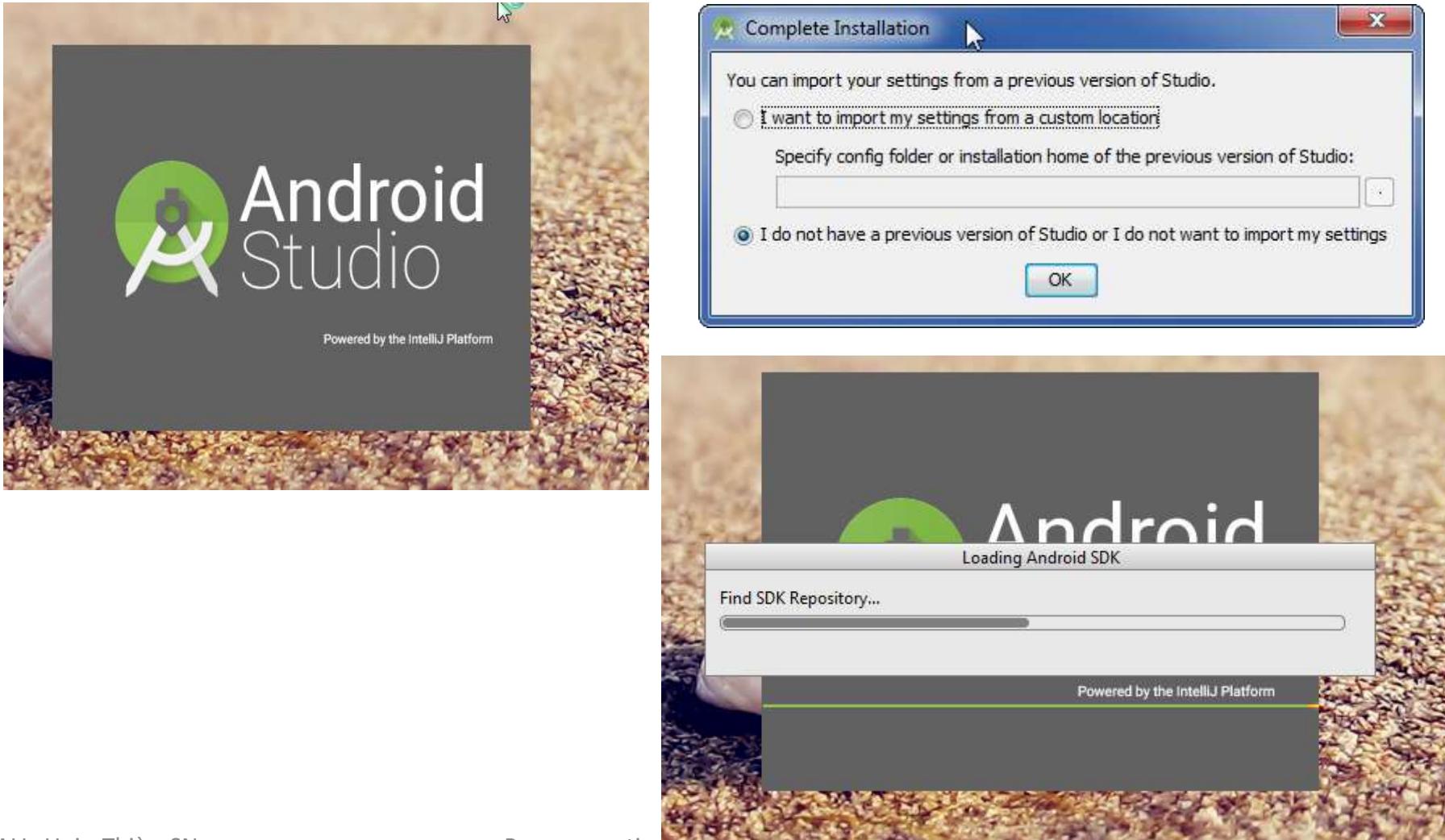
# Android Studio Installation



# Android Studio Installation



# Lancer Android Studio



# Qu'est-ce qu'une application Android?

- Un ou plusieurs écrans interactifs
- Écrit en utilisant le langage de programmation Java et XML
- Utilise le Kit de développement logiciel Android (SDK)
- Utilise les bibliothèques Android et Android Application Framework
- Exécuté par Android Runtime Virtual machine (ART)

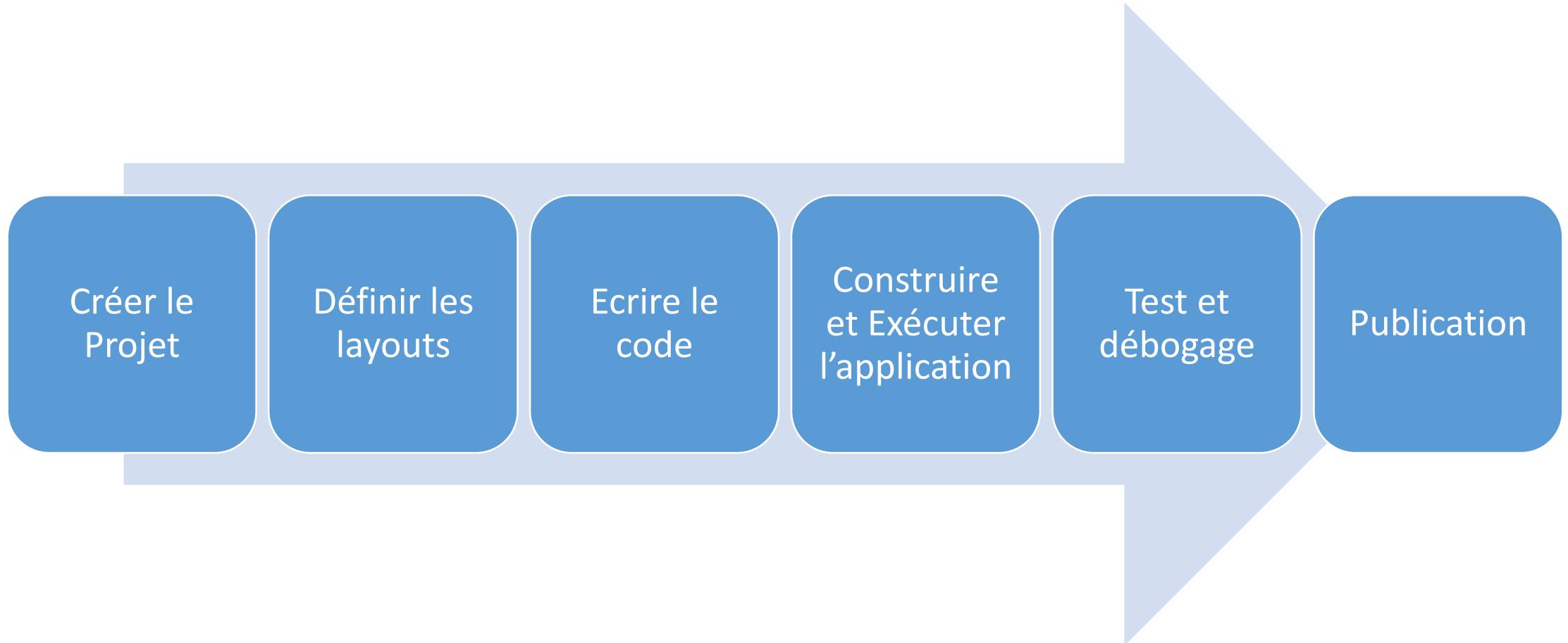
# Défis du développement Android

- Des écrans de tailles et résolutions variées
- Performance: rendez vos applications réactives et fluides
- Sécurité: protégez le code source et les données utilisateur
- Compatibilité: fonctionne bien sur les anciennes versions de plate-forme
- Marketing: comprendre le marché et vos utilisateurs

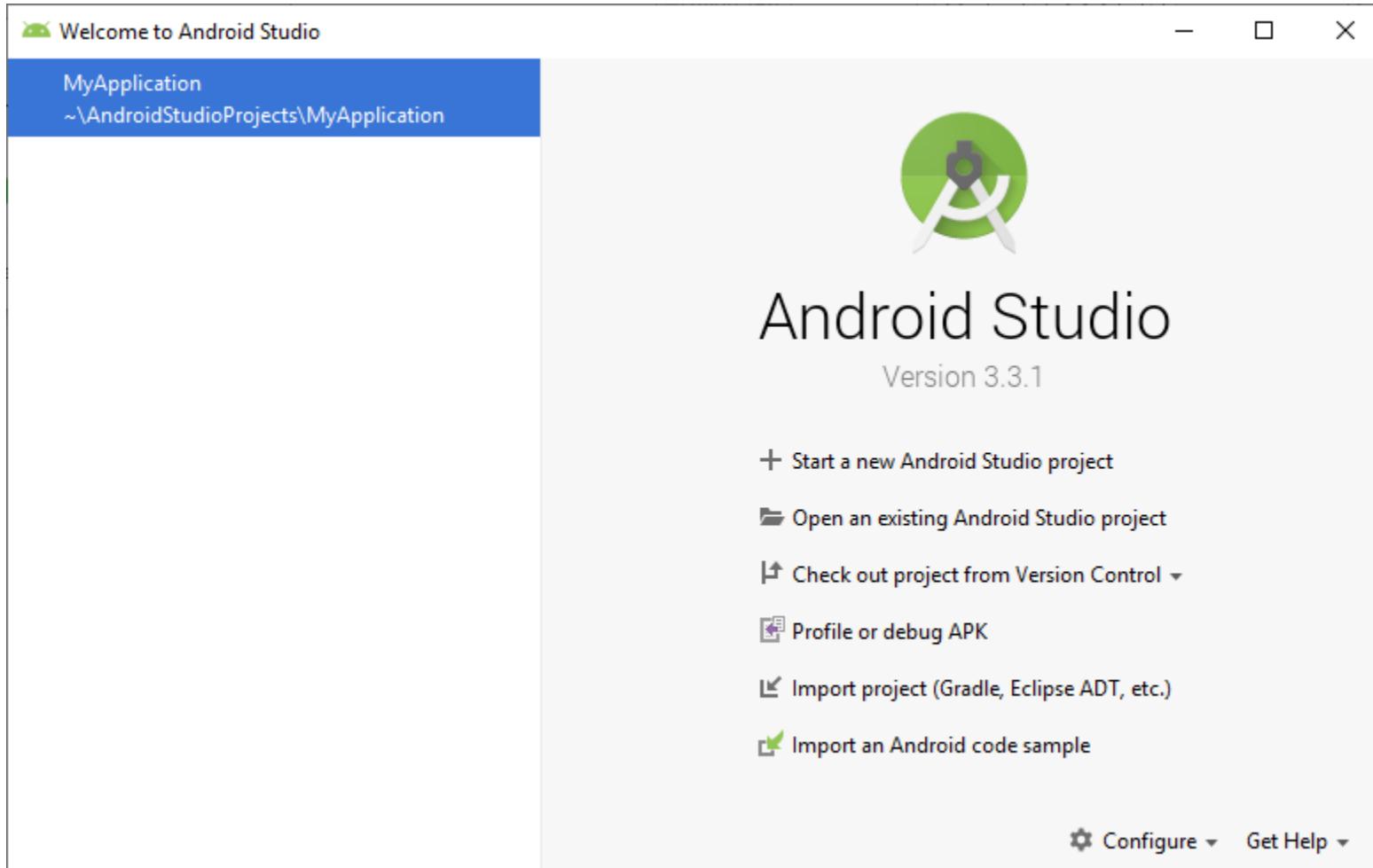
# Blocs de construction d'applications

- Ressources: mises en page, images, chaînes de caractères, couleurs au format XML et fichiers multimédias
- Composants: activités, services et classes auxiliaires sous forme de code Java
- Manifeste: informations sur l'application pour l'exécution
- Configuration de construction: versions APK dans les fichiers de configuration Gradle

# Le processus de développement

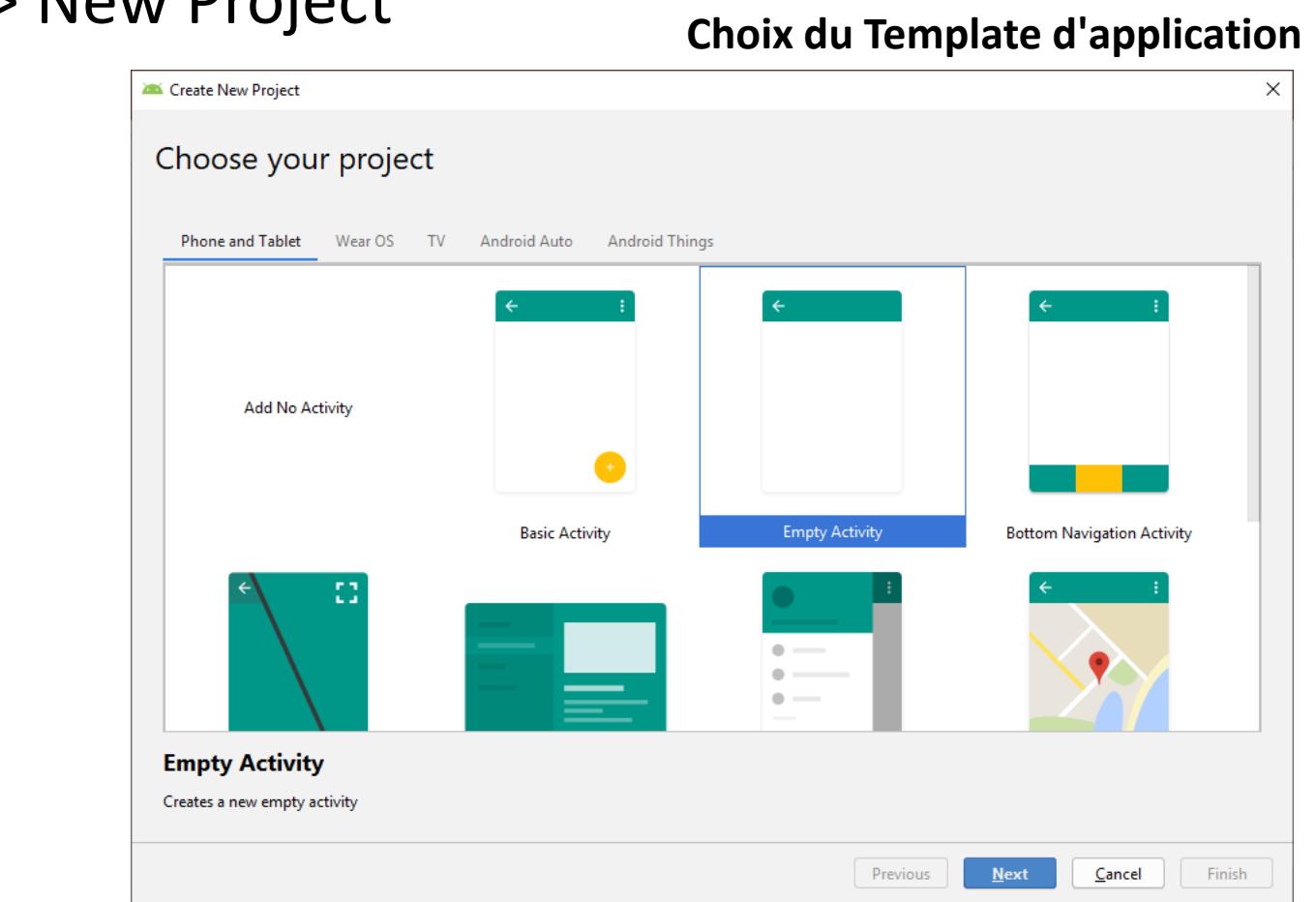
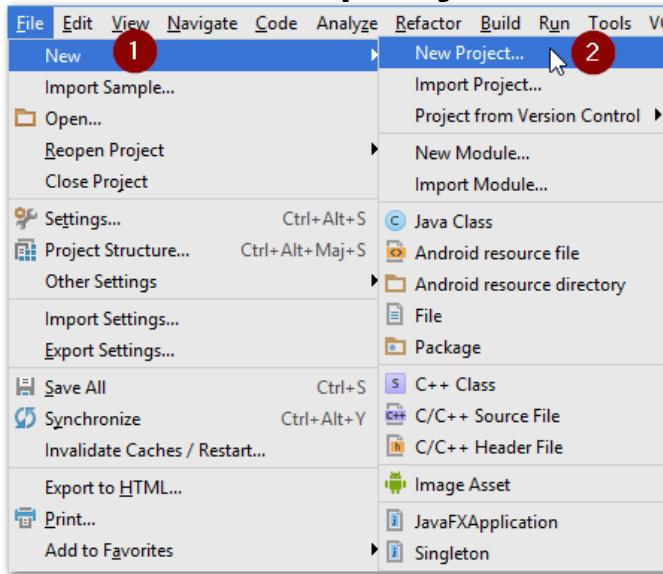


# Préparation de votre ordinateur :créer votre premier projet android

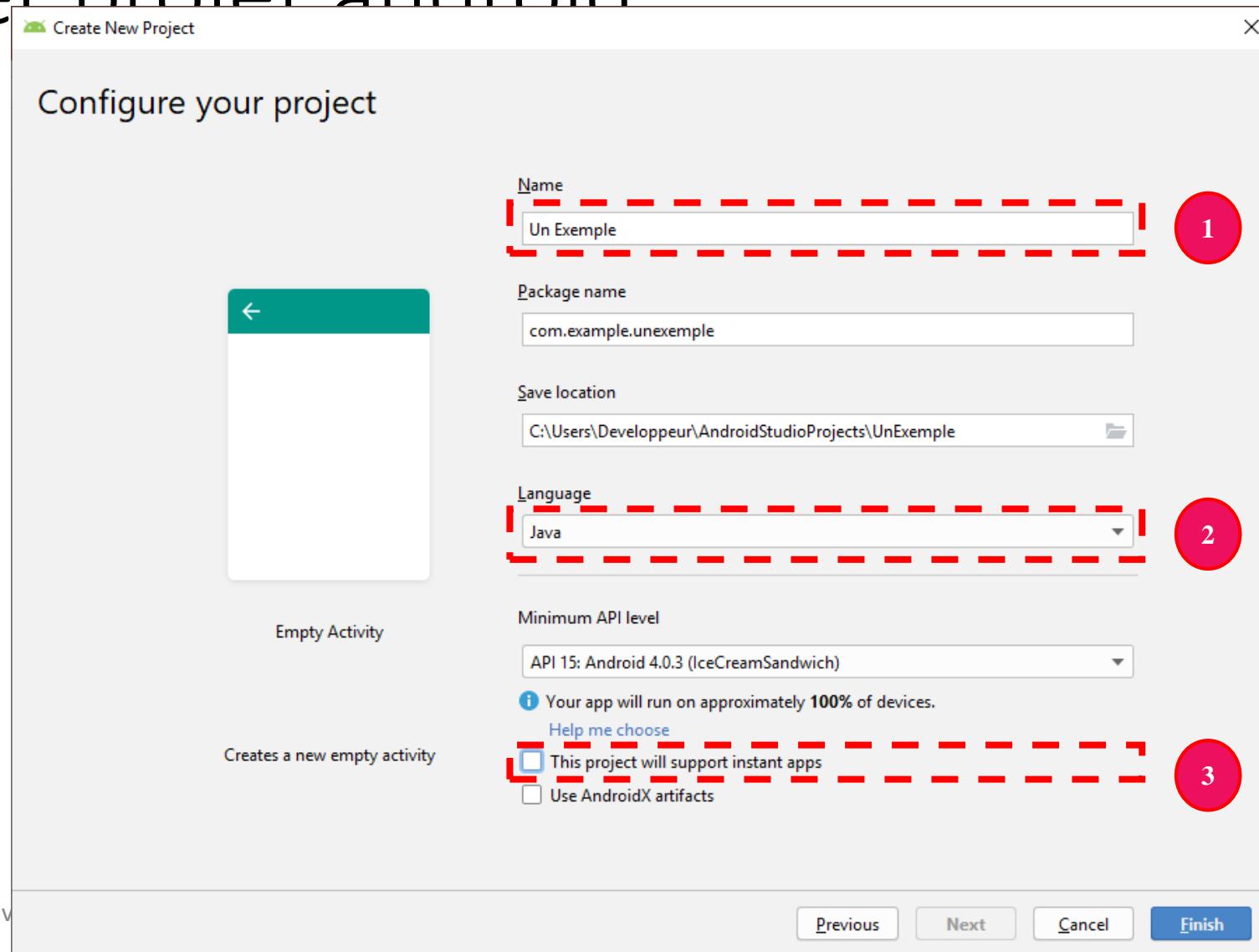


# Préparation de votre ordinateur : créer votre premier projet android

- Create a new project : File-> New Project



# Préparation de votre ordinateur : créer votre premier projet android



UnExemple [C:\Users\Developpeur\AndroidStudioProjects\UnExemple] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

UnExemple app src main res layout activity\_main.xml

1: Project

Android

app manifests java com.example.unexemple MainActivity com.example.unexemple (androidTest) com.example.unexemple (test) generatedJava res Gradle Scripts

Build Variants

2: Structure

3: Favorites

4: Layout Captures

Build Sync

Build: completed successfully at 04/04/2019 12:24

Run build C:\Users\Developpeur\AndroidStudioProjects\UnExemple

Load build

Configure build

Calculate task graph

Run tasks

Device File Explorer

8dp

Pixel 28 AppTheme 17%

Attributes

Common Ab TextView Button ImageView RecyclerView <fragment> ScrollView Switch

Text

Buttons

Widgets

Layouts

Container

Google

Legacy

Component Tree

ConstraintLayout

Ab TextView- "Hello World!"

Hello World!

Hello World!

Design Text

TODO Terminal Build Logcat Event Log

Gradle build finished in 6 s 619 ms (3 minutes ago)

Context: <no context>



1: Project

1: activity\_main.xml x 2: MainActivity.java x

Gradle

1 package com.example.unexemple;

2

3 import ...

4

5

6 public class MainActivity extends AppCompatActivity {

7

8 @Override

9 protected void onCreate(Bundle savedInstanceState) {

10 super.onCreate(savedInstanceState);

11 setContentView(R.layout.activity\_main);

12 }

13 }

14

Build Sync

Build: completed successfully at 04/04/2019 12:24

Run build C:\Users\Developpeur\AndroidStudioProjects\UnExemple

Load build

Configure build

Calculate task graph

Run tasks

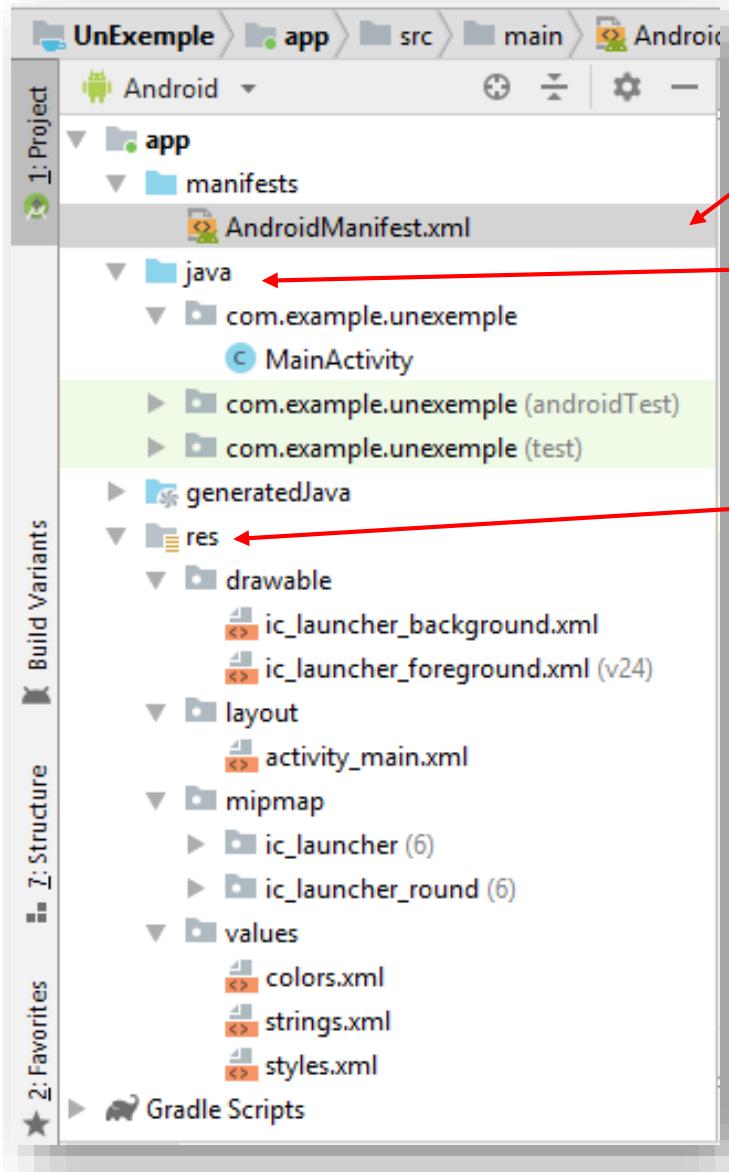
Device File Explorer

Time	Task
3 s 883 ms	Build: completed successfully
3 s 495 ms	Run build C:\Users\Developpeur\AndroidStudioProjects\UnExemple
10 ms	Load build
1 s 192 ms	Configure build
49 ms	Calculate task graph
2 s 232 ms	Run tasks

# Gradle build system

- Modern build subsystem in Android Studio
- Three build.gradle:
  - project
  - module
  - settings
- Typically not necessary to know low-level Gradle details
- Learn more about gradle at <https://gradle.org/>

# Structure d'un projet Android



Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

- Android app modules
- Library modules
- Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

All the build files are visible at the top level under **Gradle Scripts** and each app module contains the following folders:

- **manifests**: Contains the `AndroidManifest.xml` file.
- **java**: Contains the Java source code files, including JUnit test code.
- **res**: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select **Project** from the **Project** dropdown (in figure 1, it's showing as **Android**).

The project consists of three main categories of files:

- The configuration files are XML-like `Manifest.xml`, `activity_main.xml`, `string.xml` ...;
- The Java source files that implement business services, ...
- The resources that the icons and other resources

# Préparation de votre ordinateur : Créer un émulateur

- Il nécessaire d'avoir un terminal Android pour pouvoir tester son application
  - Si vous avez un terminal Android
    - Activation du mode développeur :
      - Tappoter sur le numéro du Build (Paramètre -> A propos du Téléphone)
      - Un nouveau menu est disponible (Paramètre -> Options pour les développeurs)
        - Activer le mode Débogage USB
      - Brancher votre terminal
        - Cela dépend du modèle
          - Soit en mode PTP (PHOTO dans les paramètres USB)
          - Soit en mode MTP (Multimedia dans les paramètres USB)
    - Sinon utiliser un émulateur android

UnExemple [C:\Users\Developpeur\AndroidStudioProjects\UnExemple] - ...app\src\main\AndroidManifest.xml [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

1

Android Virtual Device Manager

Your Virtual Devices

Type Name Play Store Resolution API

Pixel ...		1440 ...	28
-----------	--	----------	----

2

+ Create Virtual Device...

3

Build Variants

2: Favorites

Layout Captures

Build Sync

Build: completed successfully at 04/04/2019 1

- Run build C:\Users\Developpeur\AndroidS...
- Load build
- Configure build
- Calculate task graph
- Run tasks

TODO Terminal Build Logcat

Gradle build finished in 6 s 619 ms (19 minutes ago)

Virtual Device Configuration

Select Hardware

Choose a device definition

Category

Name	Play Store	Size	Resolution	Density
Pixel 2 XL		5,99"	1440x2880	560dpi
Nexus 6		5,96"	1440x2560	560dpi
Nexus 6P		5,7"	1440x2560	560dpi
Pixel XL		5,5"	1440x2560	560dpi
5,4" FWVGA		5,4"	480x854	mdpi
Nexus 5X		5,2"	1080x1920	420dpi
5,1" WVGA		5,1"	480x800	mdpi
Pixel		5,0"	1080x1920	420dpi
Pixel 2		5,0"	1080x1920	420dpi

4

New Hardware Profile Import Hardware Profiles

Nexus 6

1440px

5,96"

2560px

Size: large  
Ratio: long  
Density: 560dpi

Clone Device...

5

Previous Next Cancel Finish Help

TTFD CRLF UTF-8 Context: <no context>



## System Image

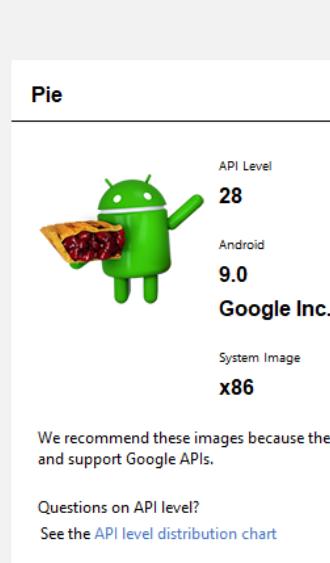
Android Studio

### Select a system image

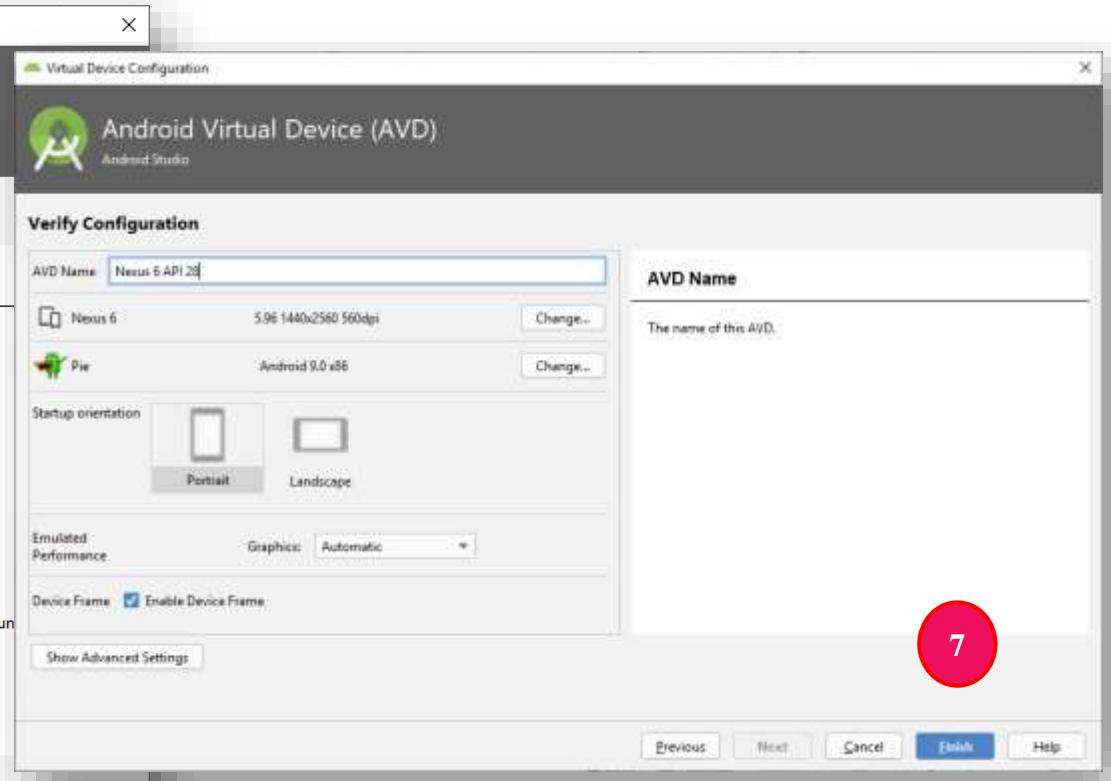
Recommended x86 Images Other Images

Release Name	API Level	ABI	Target
<a href="#">Q Download</a>	Q	x86	Android API Q (Google APIs)
<b>Pie</b>	<b>28</b>	<b>x86</b>	<b>Android 9.0 (Google APIs)</b>
<a href="#">Oreo Download</a>	27	x86	Android 8.1 (Google APIs)
<a href="#">Oreo Download</a>	26	x86	Android 8.0 (Google APIs)
<a href="#">Nougat Download</a>	25	x86	Android 7.1.1 (Google APIs)
<a href="#">Nougat Download</a>	24	x86	Android 7.0 (Google APIs)
<a href="#">Marshmallow Download</a>	23	x86	Android 6.0 (Google APIs)
<a href="#">Lollipop Download</a>	22	x86	Android 5.1 (Google APIs)

6



[Previous](#) [Next](#) [Cancel](#) [Finish](#)

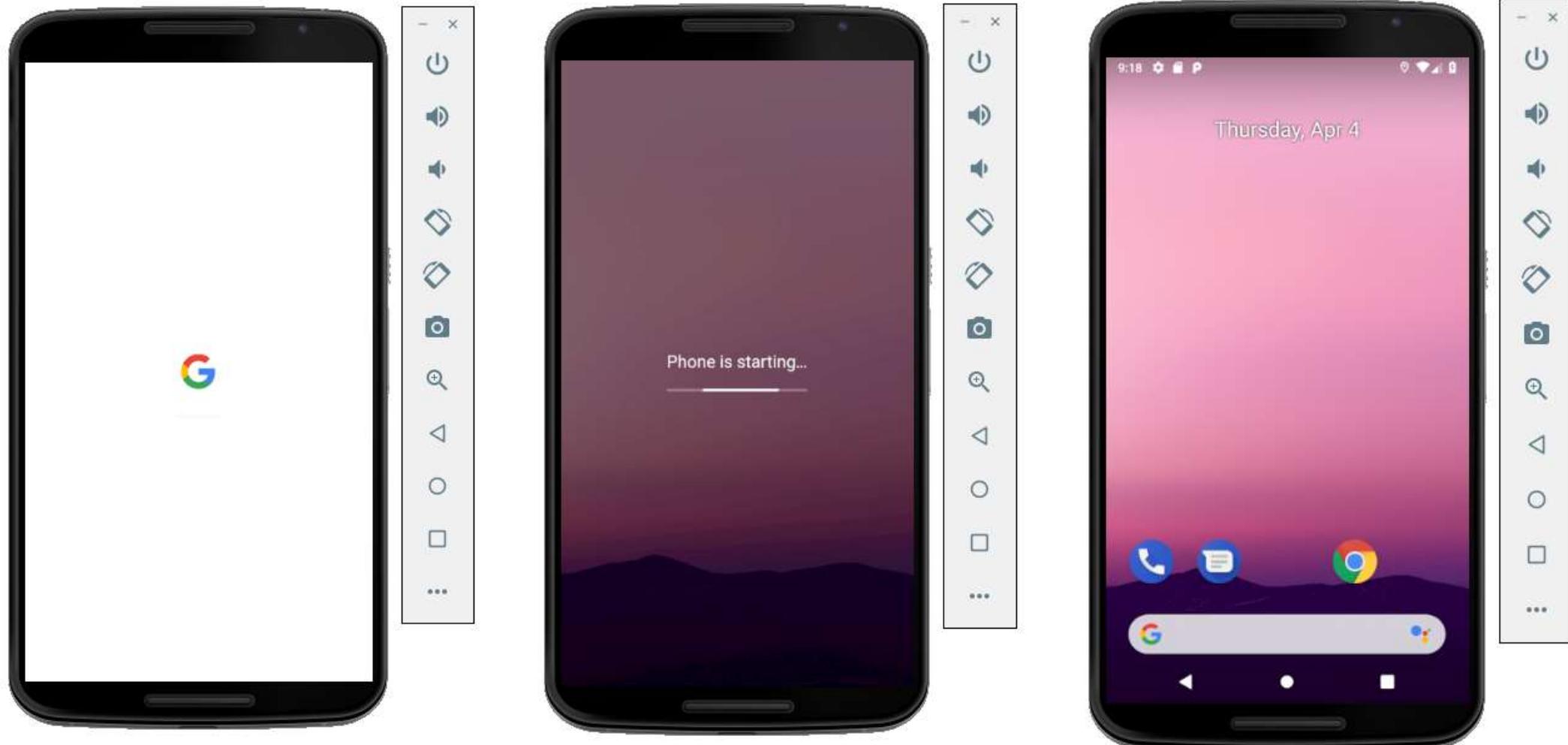


7

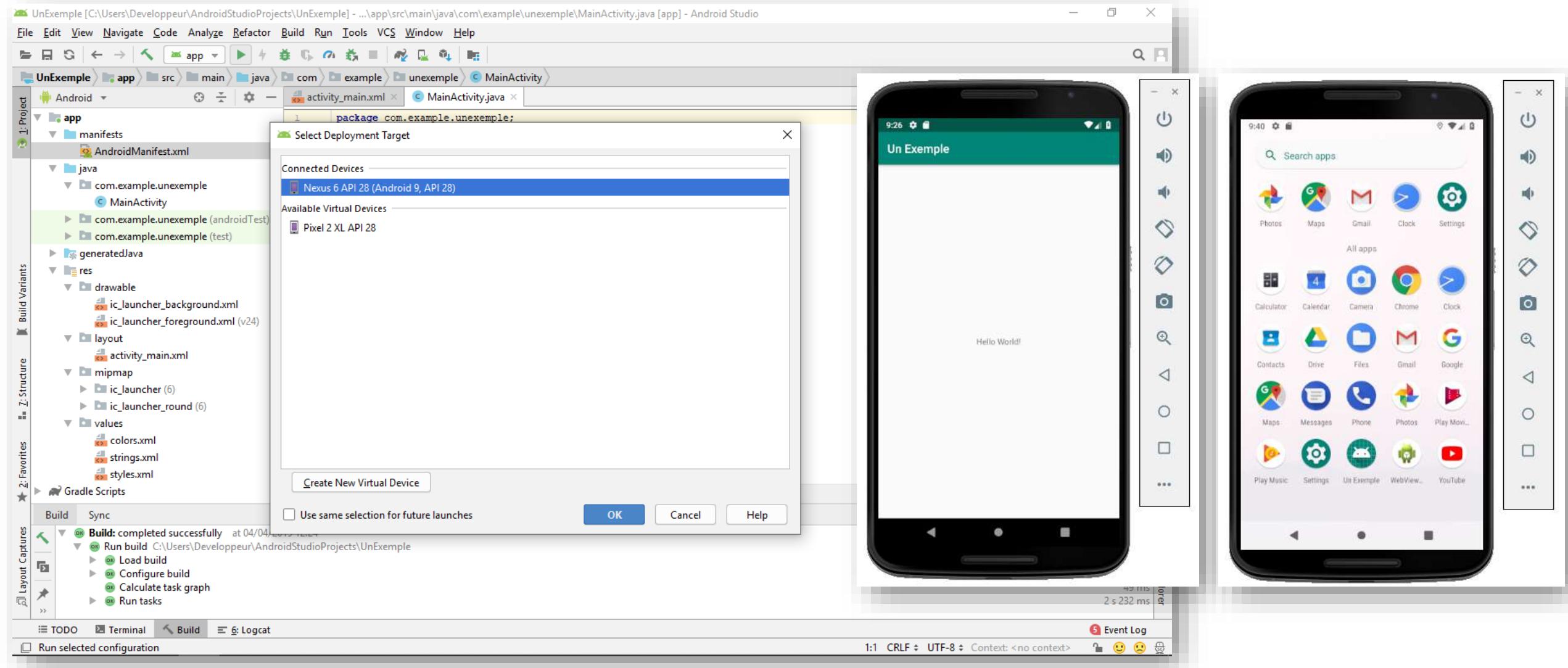


La taille initiale de 513MB va grandir au premier lancement jusqu'à plus de 1GB.

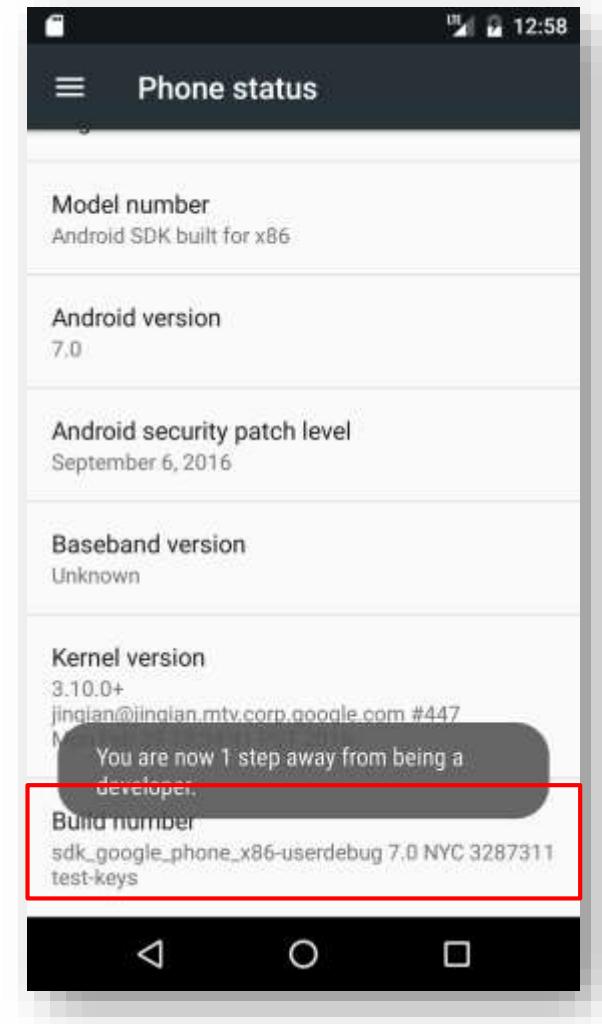
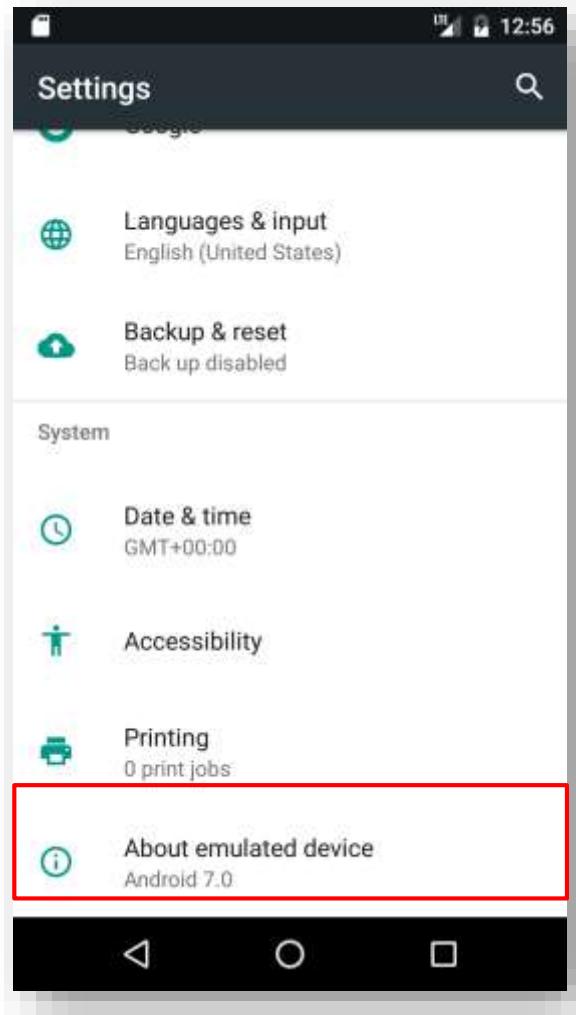
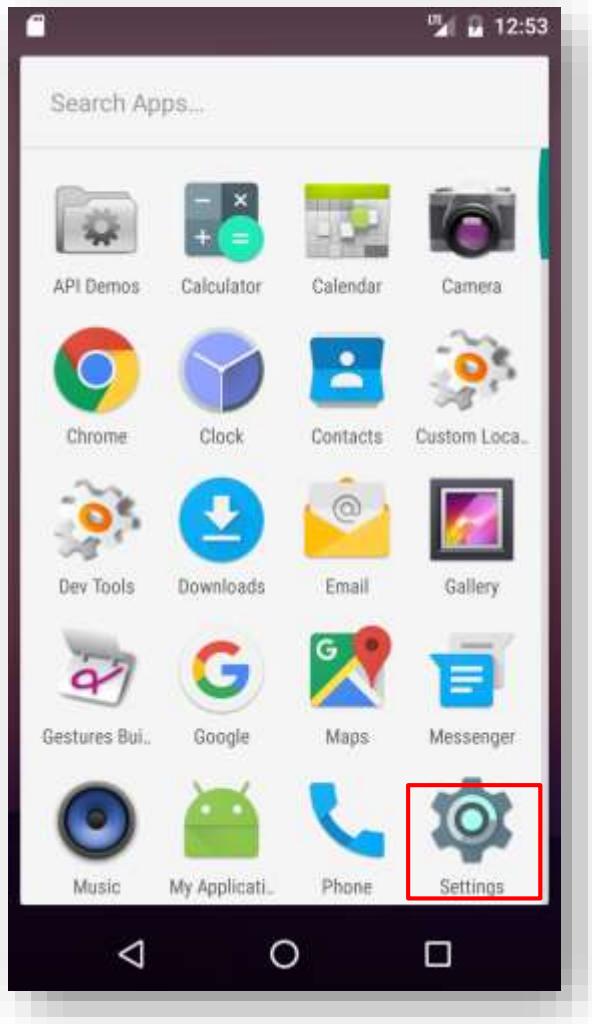
# Capture des différentes étapes du démarrage de l'emulateur



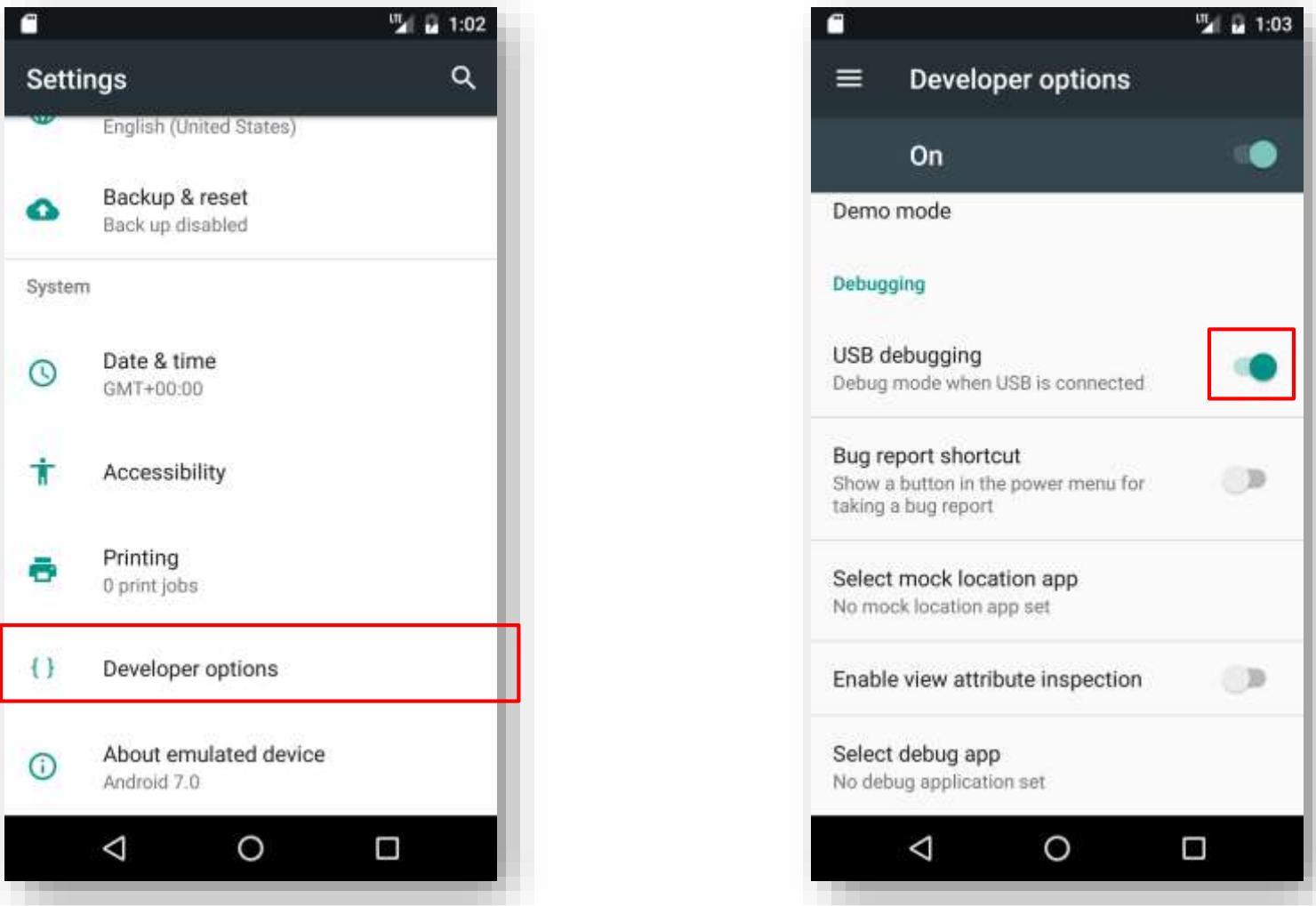
# Exécuter votre Application



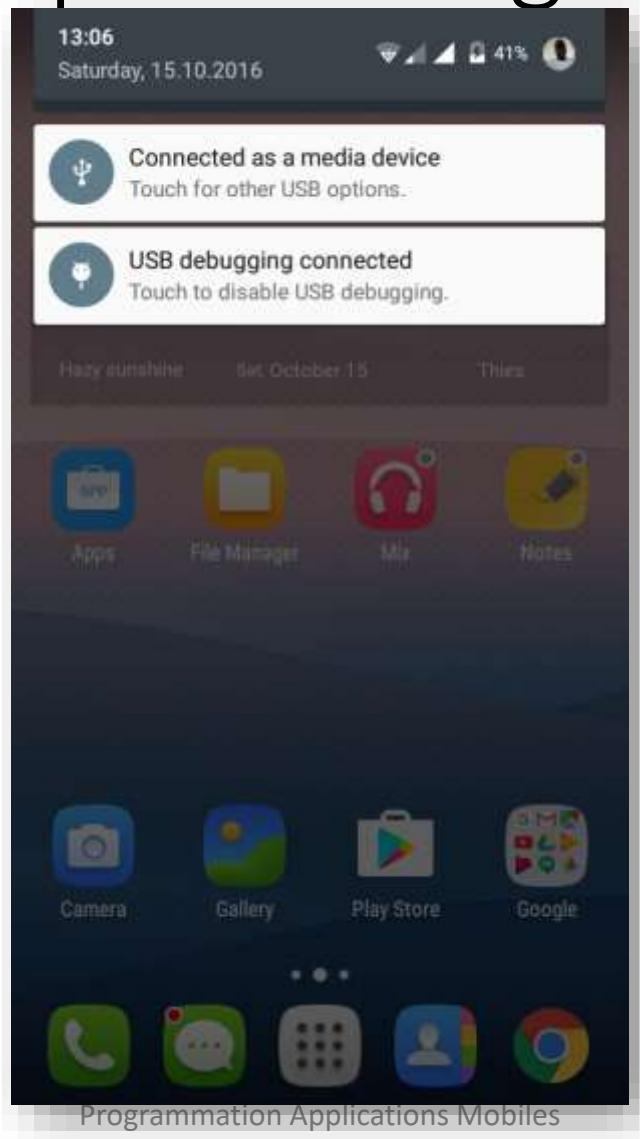
# Configurer votre téléphone Android



# Configurer votre téléphone Android



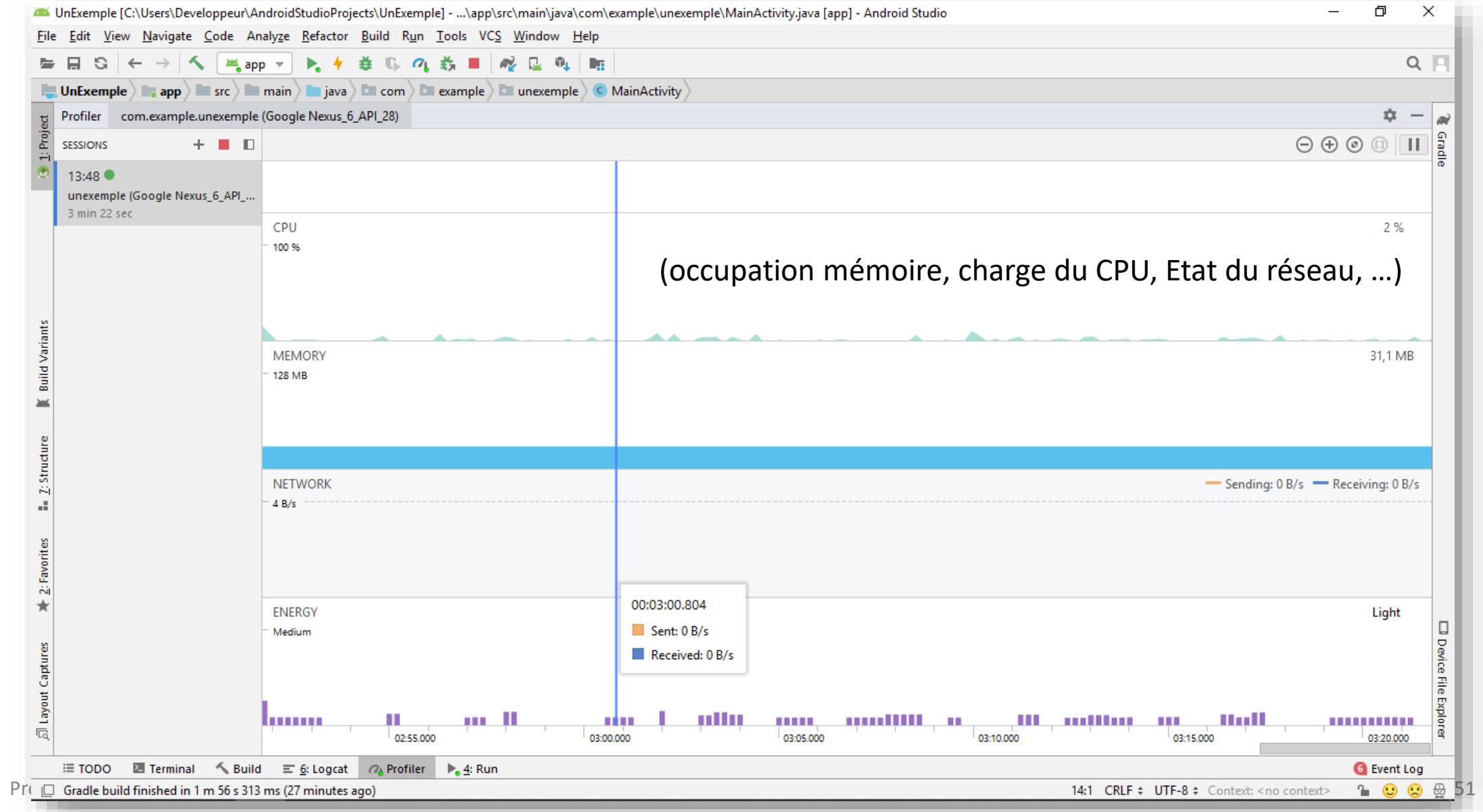
# Mon téléphone après configuration



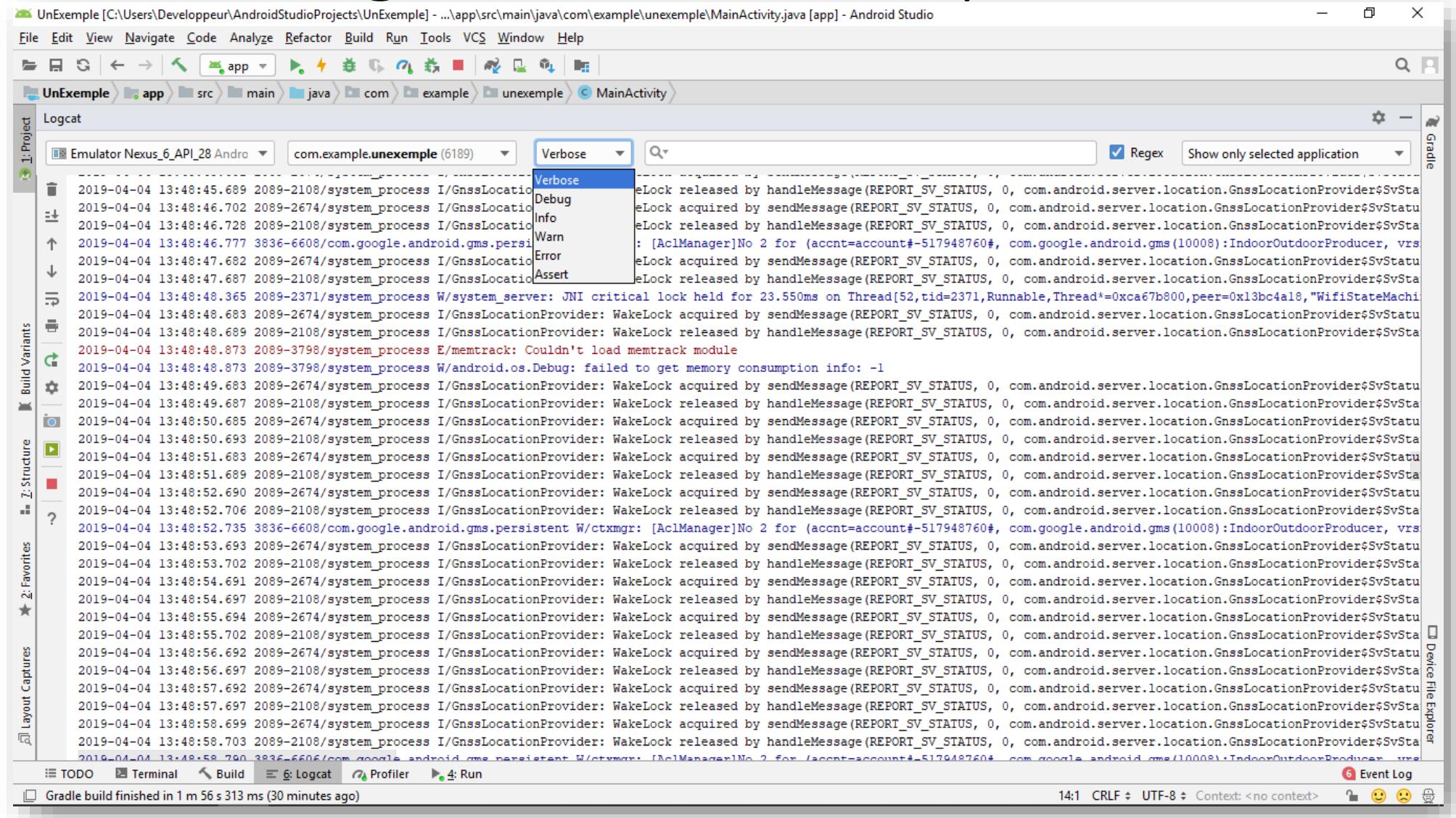
# Evaluation

- Quels sont les différentes étapes pour créer un projet Android sous Android Studio
- Rappelez le rôle des différents blocs d'Android Studio
- Quels sont les principaux types de fichiers qu'on peut trouver dans l'application Android ?
- Rappelez le rôle des différents blocs d'Android Studio

# Exercice : Logcat et Profiler Exploration



# Exercice : Logcat et Profiler Exploration

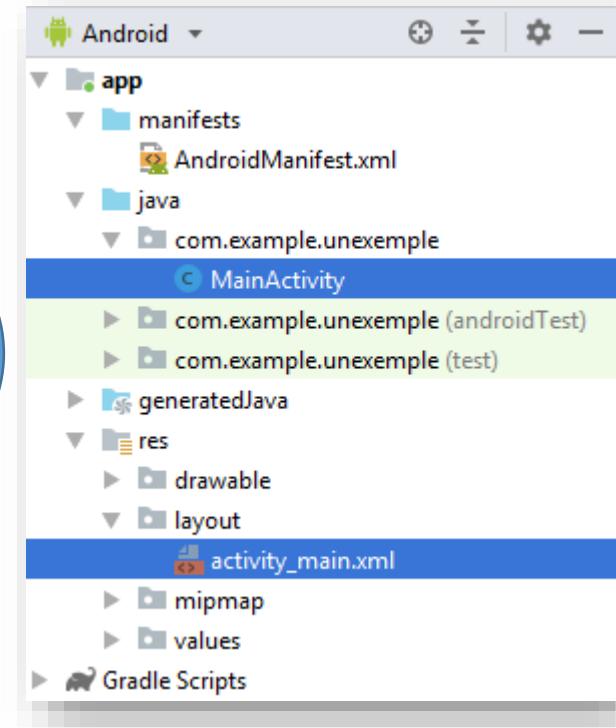
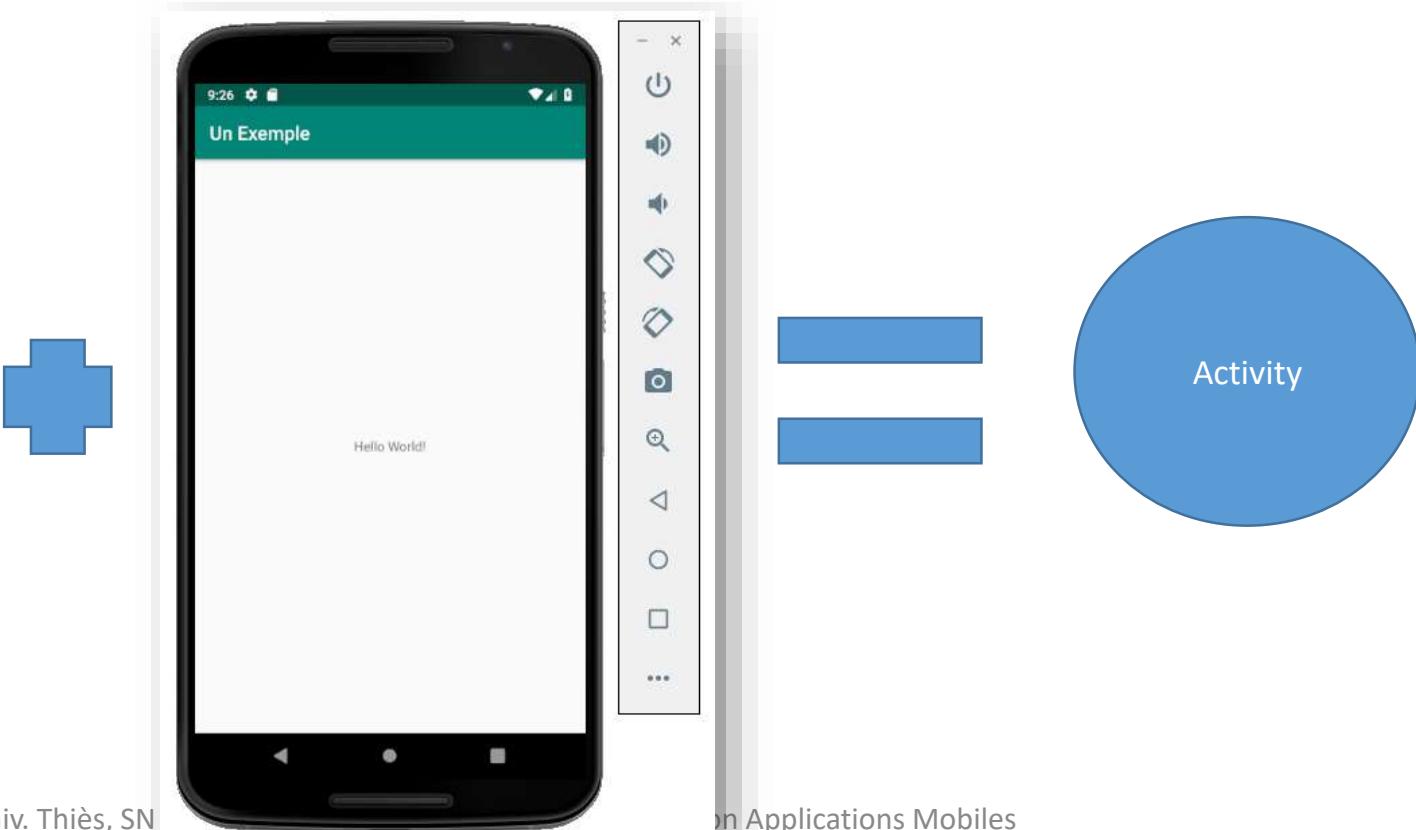


# Android application :

- Une application Android typique consiste en une ou plusieurs activités.
- Une activité est à peu près équivalente à un Windows-Form.
- Une activité montre généralement une seule interface utilisateur visuelle (GUI).
- Une seule activité (appelée principale) est choisie pour être exécutée en premier au lancement de l'application.
- Une activité peut transférer le contrôle et les données vers une autre activité via un protocole de communication interprocessus appelé Intents.

- Activity = Context + Graphic Interface

Interface(s) Graphique(s)



# Services

- Un service est un composant qui fonctionne en tache de fond pour effectuer des opérations sur une longue durée ou effectuer un travail pour les processus distants.
- Un service ne fournit pas une interface utilisateur. Par exemple, un service peut jouer de la musique en arrière-plan pendant que l'utilisateur est sur une autre application, ou peut récupérer des données sur le réseau sans bloquer l'interaction de l'utilisateur avec une activité.
- Un autre composant, comme une activité, peut démarrer un service et le laisser tourner ou bien s'enregistrer à son niveau afin d'interagir avec elle. Un service est implémenté comme une sous-classe de Service

# Intent

- Un **Intent** est un mécanisme pour décrire une action spécifique (imaginer un **Intent** comme un messager qui demande une action d'autres composants). Par exemple, il est possible de demander le lancement de l'une des applications capables de lire un fichier MP4 sans les nommer explicitement.
- L'intention est créée avec un objet **Intent** et peut être explicite ou implicite, respectivement.

# AndroidManifest.xml file

Généré par Android Studio, contient la description de l'application



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:dist="http://schemas.android.com/apk/distribution"
    package="com.example.unexemple">
    <dist:module dist:instant="true" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Le fichier Manifest présente des informations sur l'application destinées au système Android.

# Rubriques du fichier AndroidManifest

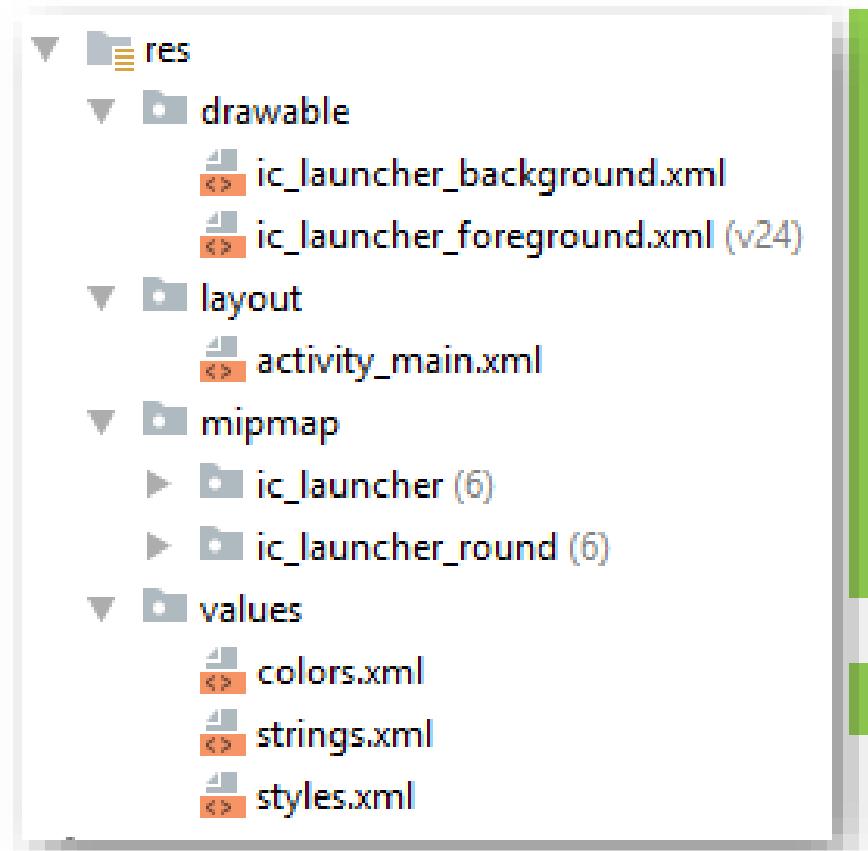
- Nom du package Java de l'application. Le nom du package est utilisé comme identifiant unique pour l'application.
- Description des composants de l'application - Activités (activités), services (services) récepteurs de radiodiffusion (de diffusion) et les fournisseurs de contenu (fournisseurs de contenu);
- Détermine le processus qui va accueillir les composants
- Déclarer les autorisations que l'application doit avoir afin d'accéder à des parties protégées de l'API et d'interagir avec d'autres applications.
- Identifier le niveau minimum de l'API Android requis par l'application.

# Rubriques du fichier AndroidManifest

- Ce fichier contient la description de l'application comprenant:
  - L'icône (**android: icon**)
  - Le nom nom de l'application (**android: label**)
  - Liste des activités qui compose l' application (**</ activity>**)
  - Description des composants de l'application - Activités (activités), services (services) récepteurs de radiodiffusion (de diffusion) et les fournisseurs de contenu (fournisseurs de contenu), permissions...

# Resources d'un projet Android

Les ressources jouent un rôle important dans un projet Android. Une ressource est un **fichier** (texte, image, son, ou autre) ou une **valeur** qui est associé à une application exécutable. Ces ressources peuvent être modifiés sans recompiler l'application et ressemble beaucoup au concept de fichiers de configuration des applications classiques. Les ressources les plus courantes incluent : des chaînes de caractères (texte sur les boutons), des couleurs (thèmes de couleurs), les images (icône de l'application, images sur l'application, ...), ...

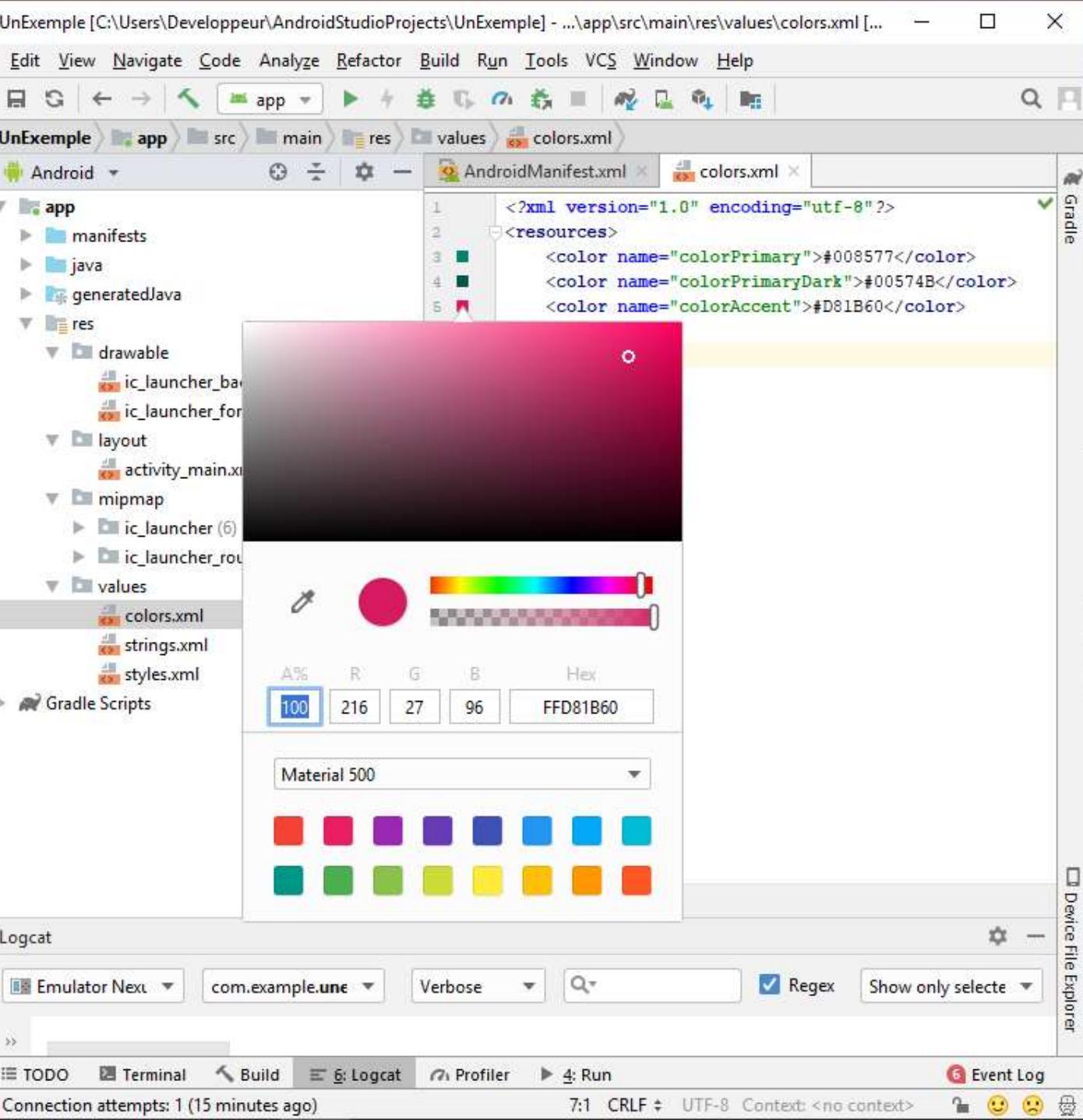


Les ressources sont stockées dans le répertoire **res/** du projet. Elles sont organisées dans des sous répertoires en fonction de leur :

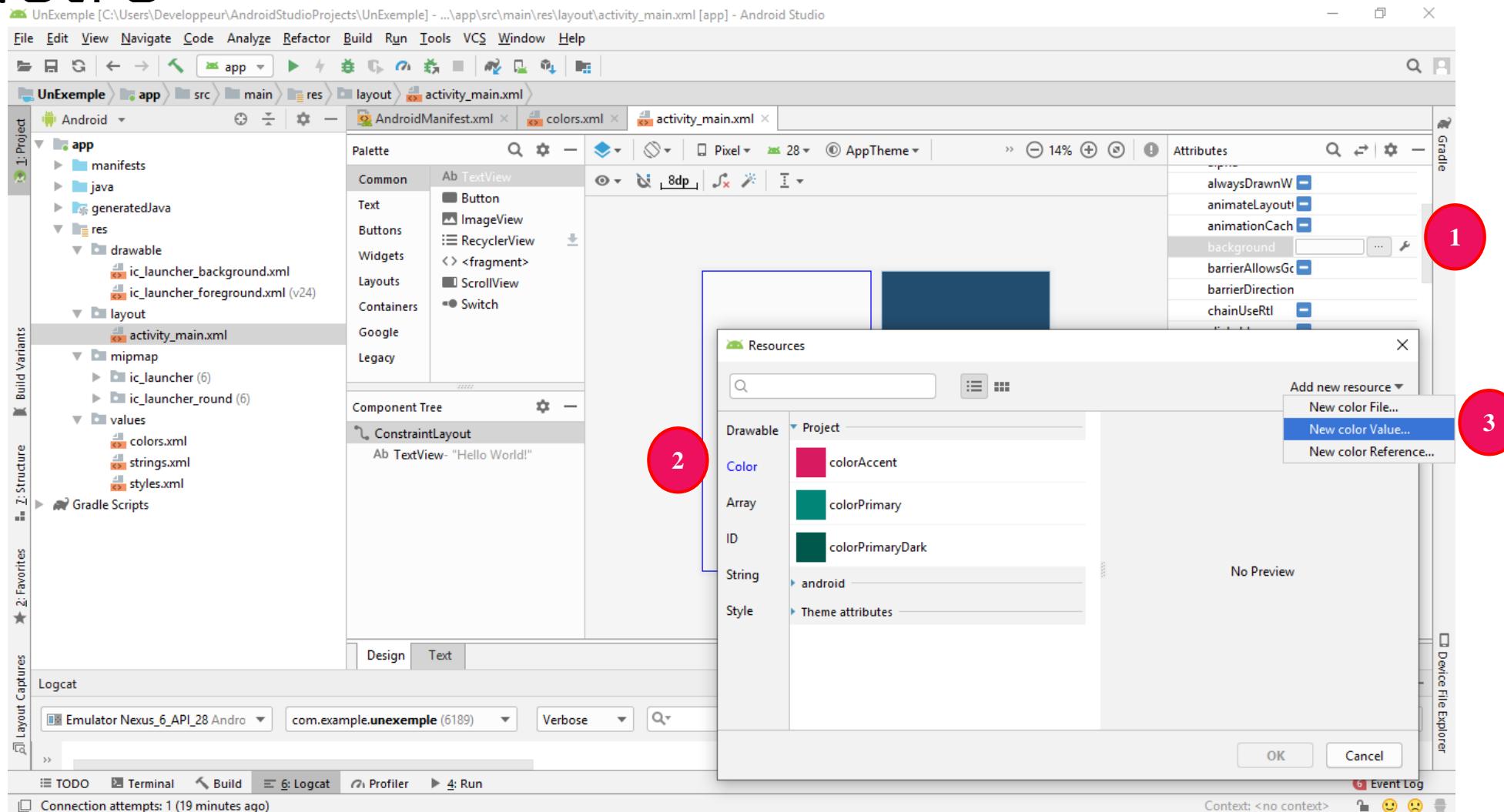
- **res/drawable** : Ce répertoire contient les images bitmaps ou XML (SVG).
- **res/layout** : Contient la définition de la présentation de l'interface de l'application (voir l'activité 2)
- **res/mipmap** : répertoire qui doit contenir l'icône de l'application avec différentes résolutions.
- **res/values** : définit des chaînes de caractères, des tableaux de chaînes de caractères (y compris le formatage de chaînes et le style), le fichier des couleurs, le fichier des styles de l'application

Demonstration avec string.xml and color.xml

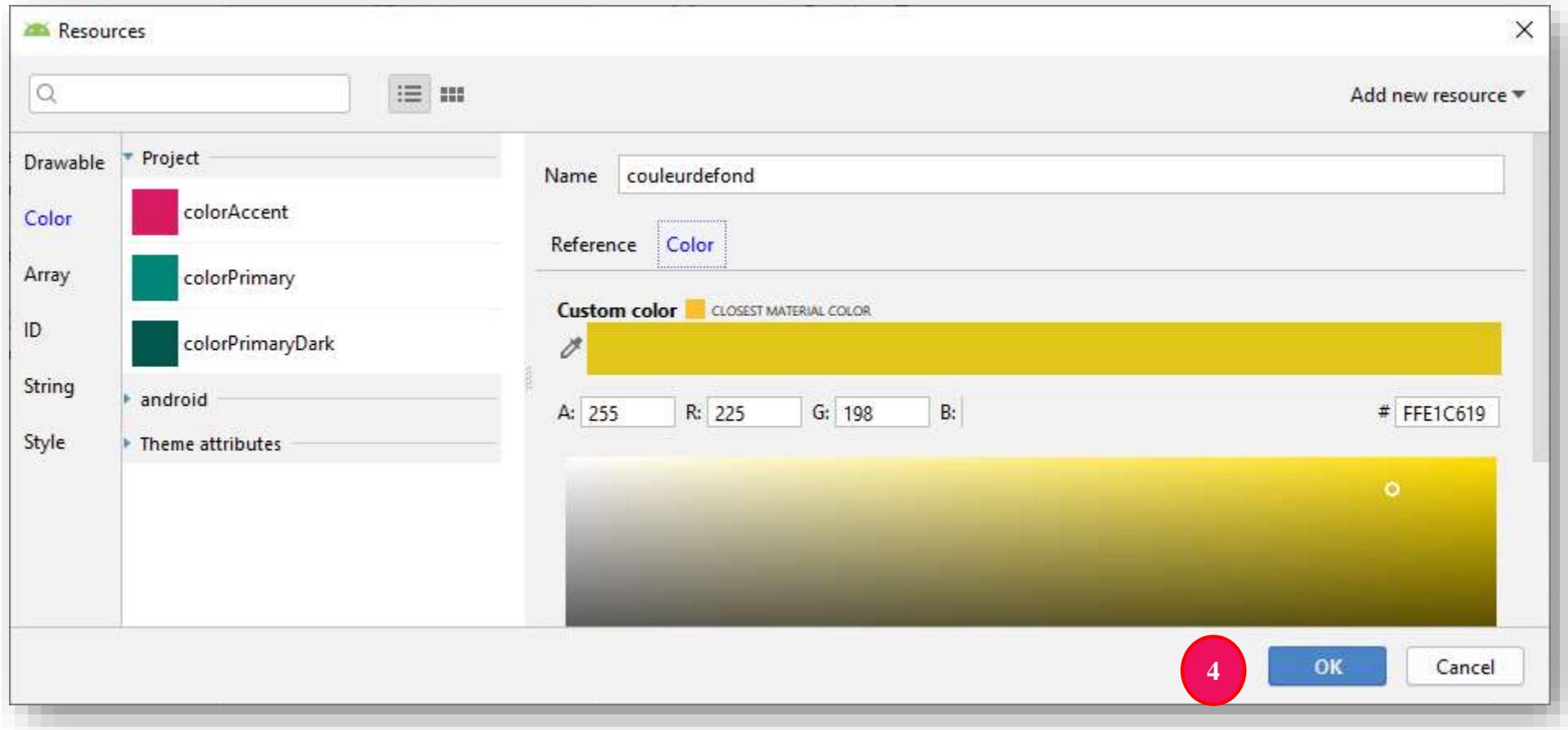
# Resources d'un projet colors.xml



# Changement de la couleur de fond de la fenêtre



# Changement de la couleur de fond de la fenêtre



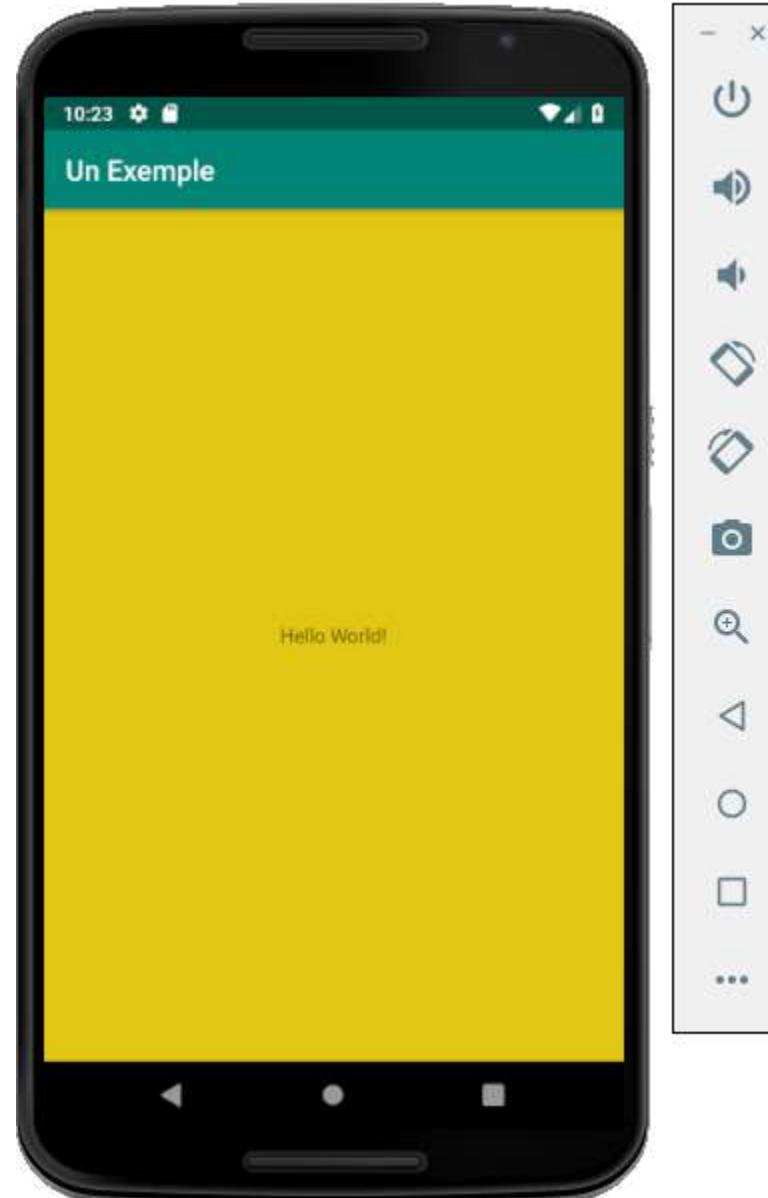
# Changement de la couleur de fond

The screenshot shows the Android Studio interface with the following details:

- Project Tab:** Shows the project structure with `app` selected.
- Layout Tab:** Shows the `activity_main.xml` layout file.
- Design View:** Displays a yellow square with a blue border, representing the background color.
- Attributes Panel:** Shows the `background` attribute set to `@color/couleurdefond`.
- Color Chooser:** A color palette is open, showing a yellow square highlighted as the selected color.
- Code Editor:** The `colors.xml` file is open, showing the definition of the color `couleurdefond`.
- Logcat Tab:** Shows logs for the Emulator Nexus\_6\_API\_28.
- Bottom Bar:** Shows tabs for `TODO`, `Terminal`, `Build`, `Logcat`, `Profiler`, `Run`, and `Event Log`.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#008577</color>
    <color name="colorPrimaryDark">#00574B</color>
    <color name="colorAccent">#D81B60</color>
    <color name="couleurdefond">#E1C619</color>
</resources>
```

# Changement de la couleur de fond de la fenêtre



# Creation d'une ressource de type string dans string.xml

The screenshot illustrates the process of creating a string resource in Android Studio:

- 1**: In the Design tab of the Layout Editor, a yellow `ConstraintLayout` contains a single `TextView` with the text "Hello World!".
- 2**: In the `strings.xml` editor, a new string resource named "message" is being created with the value "Salut tout le monde !!!".
- 3**: A context menu is open over the `activity_main.xml` layout, showing options like "Add new resource", "Edit", "Delete", etc.
- 4**: The "OK" button is highlighted in the "New String Value Resource" dialog.

The final result is shown on the right, where the app's screen displays the text "Hello World!".

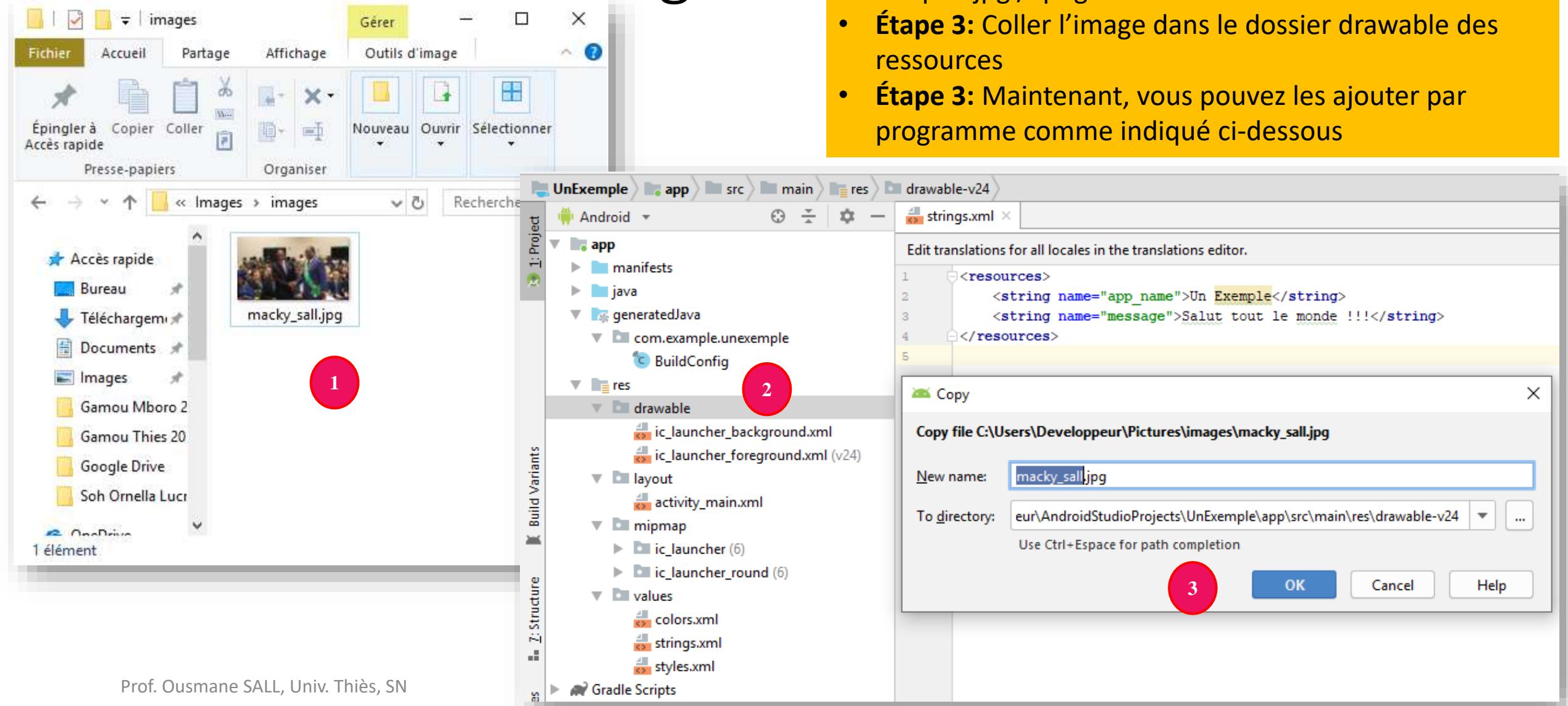
Code in `strings.xml`:

```
<resources>
    <string name="app_name">Un Exemple</string>
    <string name="message">Salut tout le monde !!!</string>
</resources>
```

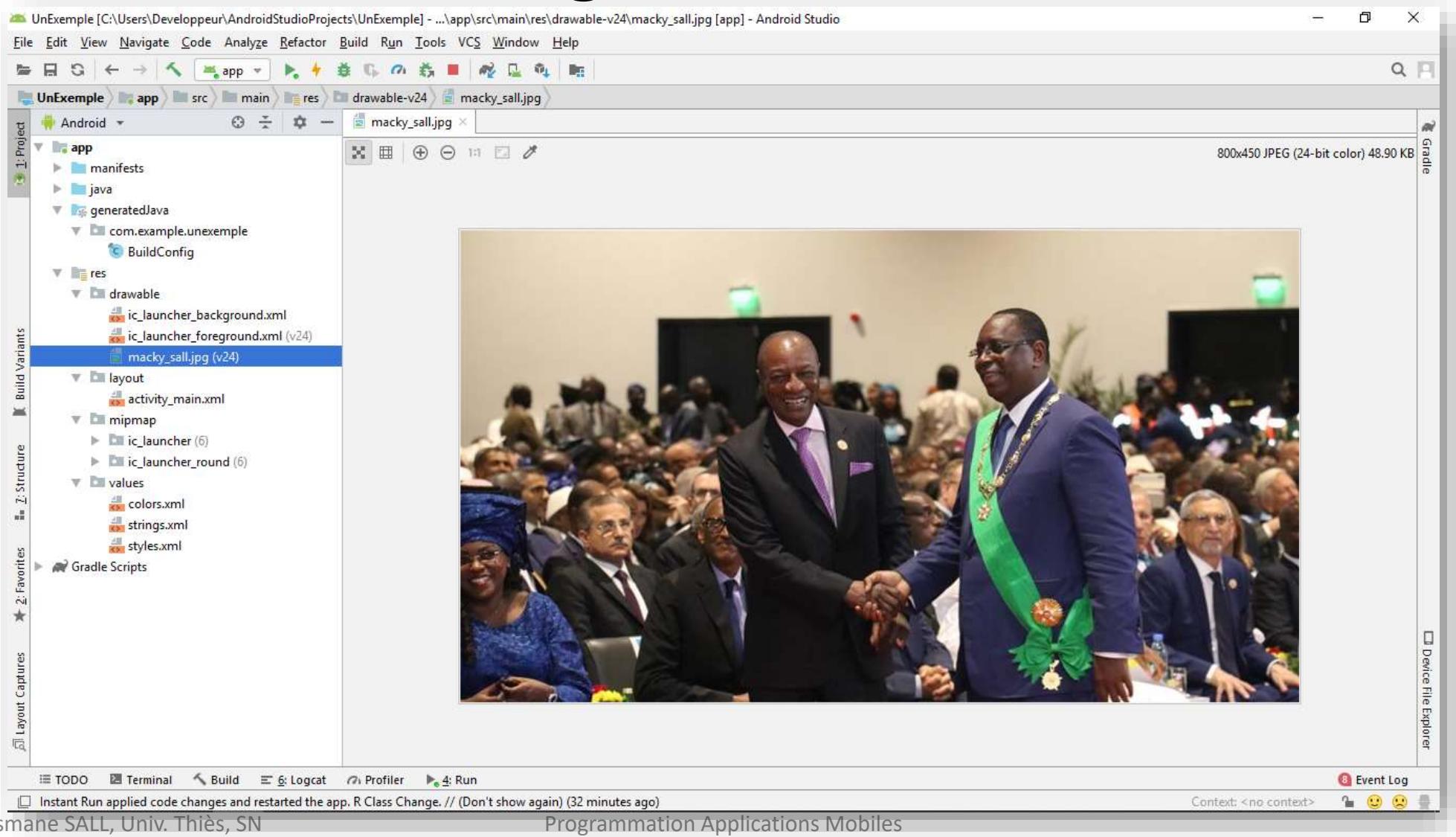
Android Studio interface elements visible include the Project, Build Variants, Favorites, Logcat, TODO, Terminal, Build, Logcat, Profiler, Run, Event Log, and Context menus.

# Utilisation des images

- **Étape 1:** renommez l'image avec uniquement des minuscules et un trait de soulignement (\_) uniquement
- **Étape 2:** L'image peut être dans n'importe quel format tel que .jpg / .png
- **Étape 3:** Coller l'image dans le dossier drawable des ressources
- **Étape 3:** Maintenant, vous pouvez les ajouter par programme comme indiqué ci-dessous



# Utilisation des images



UnExemple [C:\Users\Developpeur\AndroidStudioProjects\UnExemple] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

UnExemple app src main res layout activity\_main.xml

macky\_sall.jpg activity\_main.xml

1: Project

Android

app manifests java generatedJava com.example.unexemple BuildConfig res drawable ic\_launcher\_background.xml ic\_launcher\_foreground.xml (v24) macky\_sall.jpg (v24) layout activity\_main.xml mipmap ic\_launcher (6) ic\_launcher\_round (6) values colors.xml strings.xml styles.xml Gradle Scripts

Build Variants

2: Structure

Layout Captures

Favorites

Device File Explorer

Palette

Common Text Buttons Widgets Layouts Containers Google Legacy

Pixel 28 19% 8dp

Attributes

id layout\_width layout\_height Constraints Layout\_Margin Padding Theme background context actionBarNavMode addStatesFromChildren alpha alwaysDrawnWithCache animateLayoutChanges animationCache barrierAllowsGoneWidget barrierDirection chainUseRtl clickable clipChildren clipToPadding constraintSet constraint\_referenced\_ids contentDescription descendantFocusability drawingCacheQuality duplicateParentState

Pick a Resource @color/couleur

>MainActivity

Design Text

TODO Terminal Build Logcat Profiler Run

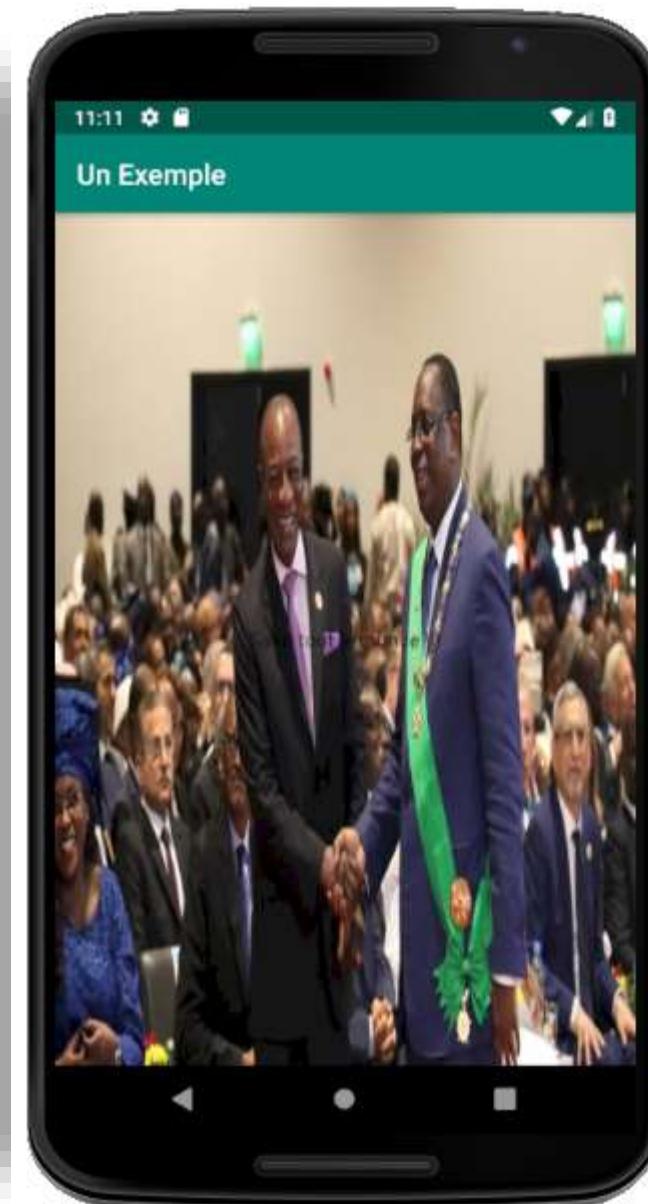
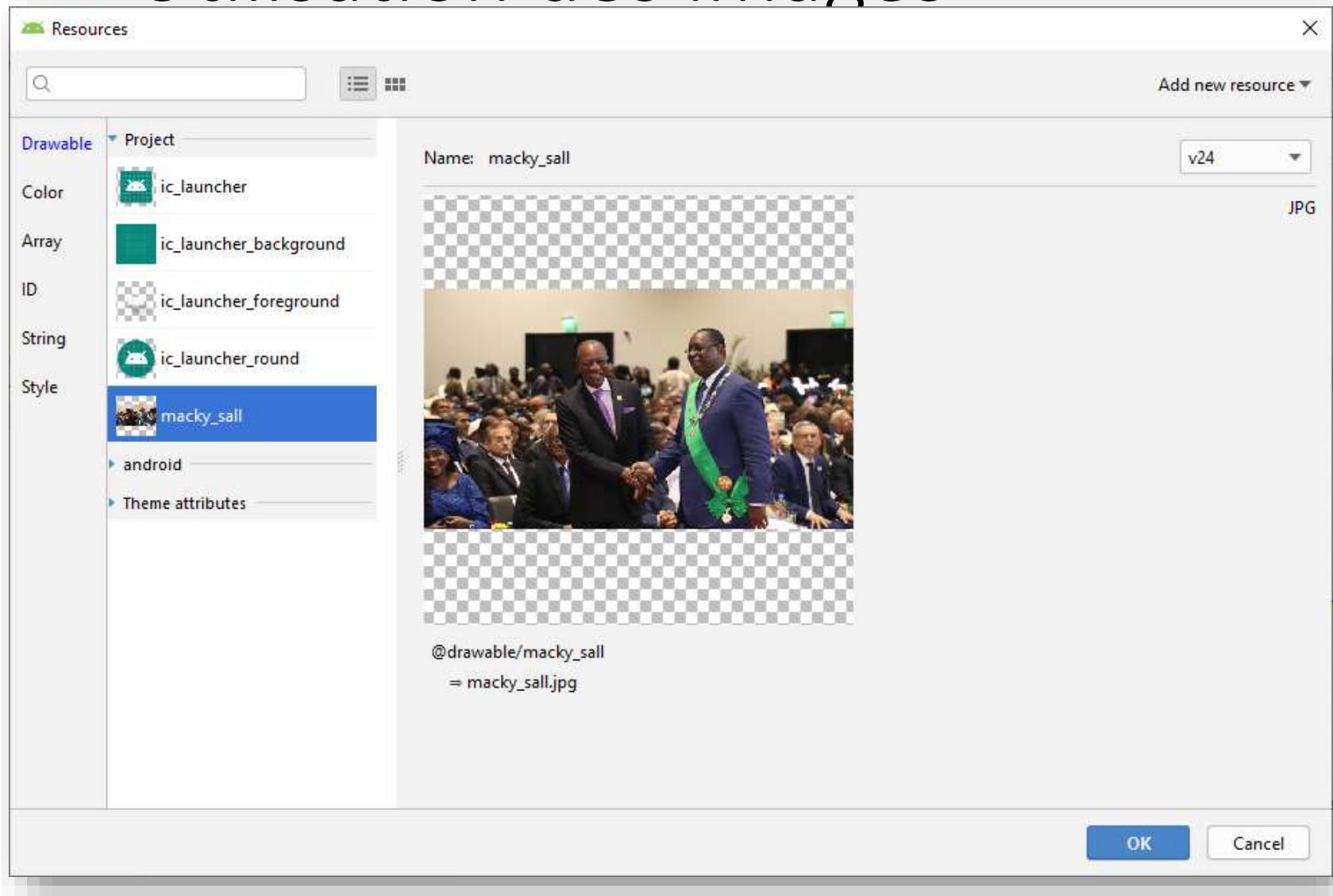
Event Log

Instant Run applied code changes and restarted the app. R Class Change. // (Don't show again) (38 minutes ago)

6 chars 1:15 Context: <no context>

Smiley Face

# Utilisation des images



# Utilisation des images: changer l'icône de l'application

The screenshot shows the Android Studio interface with the project 'UnExemple' open. The left sidebar displays the project structure under 'app'. In the main editor area, the file 'AndroidManifest.xml' is open, showing XML code for the application manifest. A yellow highlight covers the section of code where the application icon is defined:

```
<manifest>
    ...
    <application
        android:allowBackup="true"
        android:icon="@mipmap/senegal"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        ...
    </application>
</manifest>
```

The 'Text' tab of the editor is selected. To the right of the editor, a smartphone screen displays the home screen of an Android device. The app's icon, which has been changed to a Senegalese flag, is visible among other system icons like Photos, Maps, Gmail, and Clock.

# La classe R

- C'est une classe générée par Eclipse qui permet à l'application d'accéder aux ressources
- Elle contient des classes internes dont les noms correspondent aux types de ressources (id, drawable, layout ...)
- Elle est constituée à partir des fichiers placés dans les sous répertoires du répertoire res
- Une propriété est créée pour :
  - Chaque image placé dans drawable-xxxx
  - Chaque identificateur défini dans des fichiers XML (objets d'interface, constantes)
  - Chaque fichier placé dans les répertoires xml , raw ...

# Utilisation des ressources

- Référencement d'une ressource dans un fichier xml. La forme générale est :  
"@type/identificateur"
- Par exemple : @string/machaine Fait référence à une chaîne contenue dans un fichier XML placé dans le répertoire **res/values** et définie comme suit :

```
<resources>
    ...
    <string name="machaine">Contenu de cette chaîne</string>
    ...
</resources>
```

- Référencement d'une ressource dans le code. La forme générale est : R.type.identificateur
  - Par exemple : R.string.machaine
  - Fait référence à la même chaîne

# Resources classe

- Permet l'accès aux ressources répertoriées dans `R`
- On obtient une instance de cette classe par `getResources()` de l'activité
- Principales méthodes de la classe `Resources` (le paramètre est un identifiant défini dans `R` de la forme `R.type.nom`) :
  - `boolean getBoolean(int)`
  - `int getInteger(int)`
  - `int[] getArray(int)`
  - `String getString(int)`
  - `String[] getStringArray(int)`
  - `int getColor(int)`
  - `float getDimension(int)`
  - `Drawable getDrawable(int)`
- Exemple : `String titre = getResources().getString(R.string.ma_chaine);`

# Utilisation des ressources

- Accès aux ressources dans l’application
  - Mise en place de l’interface principale
    - `setContentView(R.layout.nom_du_fichier_xml);`
  - Mise en place d’interfaces supplémentaires
    - Par les classes `LayoutInflater` ou `MenuItemInflater`
  - Accès direct à une valeur ou à une ressource :
    - `String titre = getResources().getString(R.string.texte_titre);`
    - `Drawable monImage = getResources().getDrawable(R.drawable.nom_de_l_image)`

# Les chaines

- Les chaines constantes de l'application sont situées dans **res/values/strings.xml**. Voici un exemple:

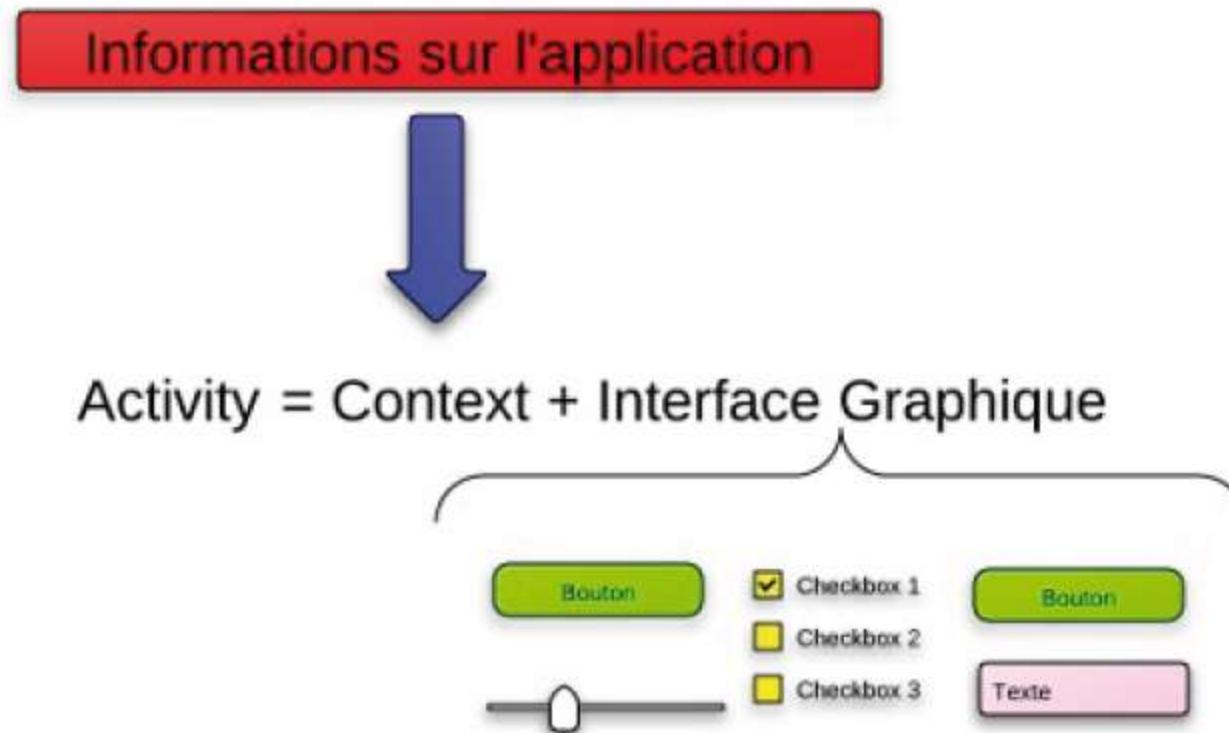
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello Hello CFPP !</string>
    <string name="app_name">AndroJF</string>
</resources>
```

- La récupération de la chaîne se fait via le code:

```
Resources res = getResources();
String hw = res.getString(R.string.hello);
```

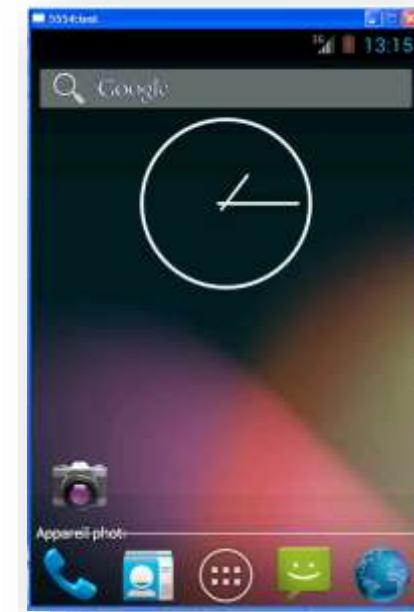
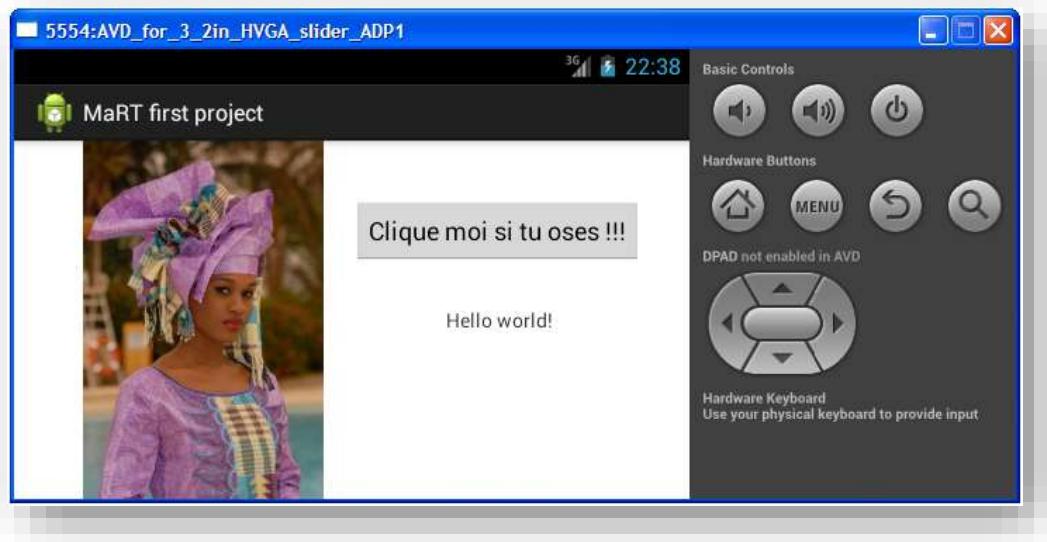
# Application Android

- Une activité = un programme + une interface



# Activités Android

- Partie importante de l'interface graphique Android
- Peut être considéré comme une «fenêtre»

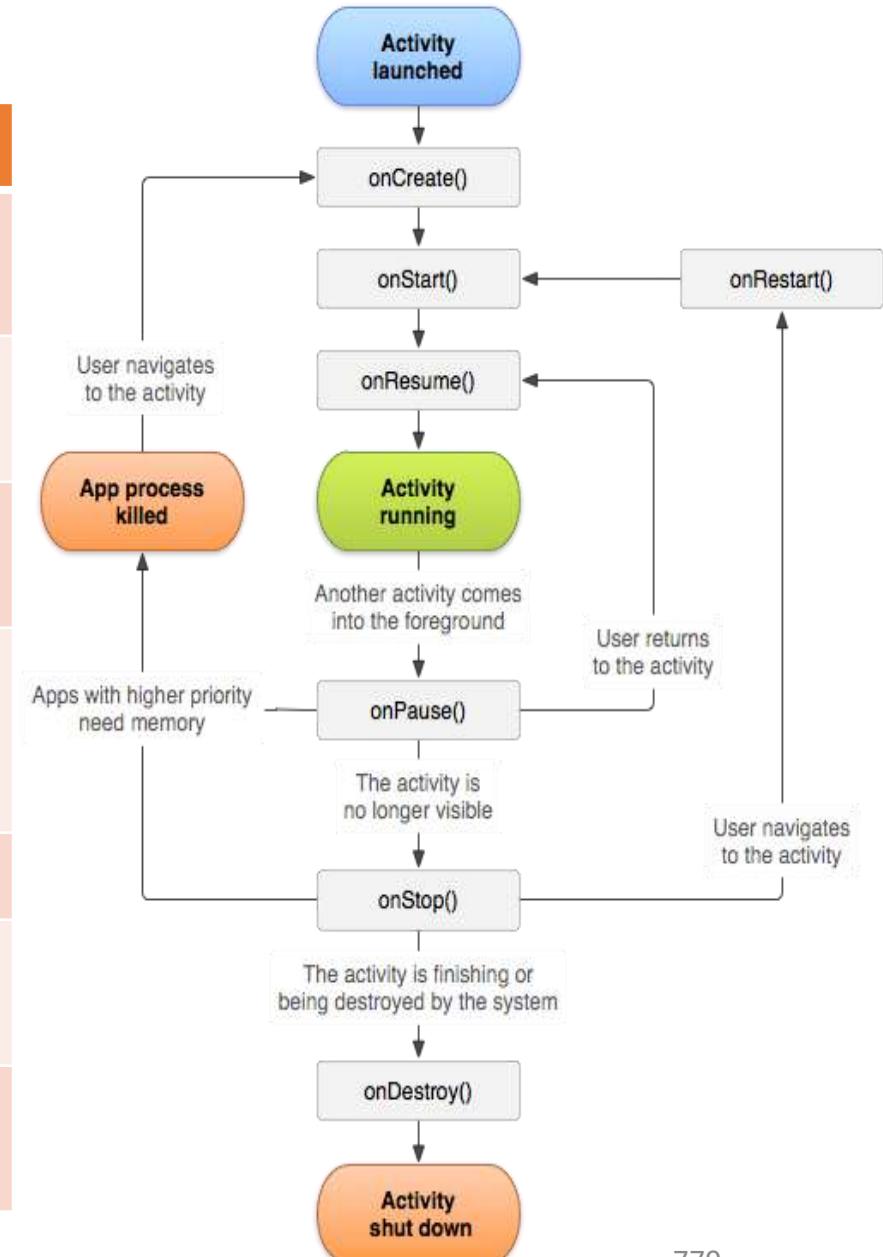


# Android Activity

- Une activité est le composant d'application qui fournit une fenêtre (un écran) à travers laquelle les utilisateurs peuvent interagir et initier une action, telles que la composition d'un numéro de téléphone, envoyer un e-mail, visionner une vidéo, etc.
- Les activités sont un élément fondamental dans le développement d'applications Android. Ils peuvent exister dans un certain nombre d'états différents. Le cycle de vie de l'activité commence par son instantiation et se termine par sa destruction, et passe par de nombreux états intermédiaires.
- Quand une activité change d'état, la méthode de l'événement du cycle de vie approprié est appelée afin d'avertir l'activité du changement imminent de son état. Ces callbacks lui permettent d'exécuter du code afin de l'adapter à ce changement.

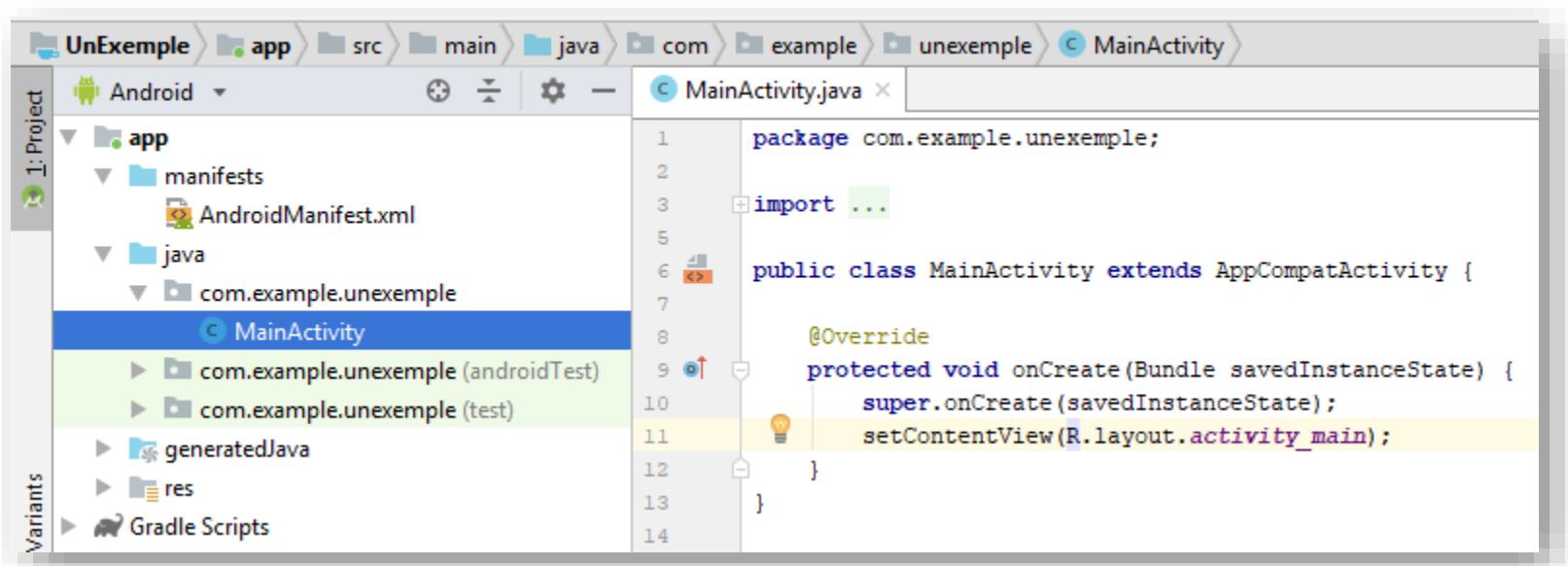
# Cycle de vie d'une activité

Callback	Description
<b>onCreate()</b>	Cette fonction est la première qui est exécutée lorsque l'activité est créée pour la première fois.
<b>onStart()</b>	Ce callback est appelé lorsque l'activité devient visible à l'utilisateur.
<b>onResume()</b>	Cette méthode est appelée lorsque l'utilisateur commence à interagir avec l'application.
<b>onPause()</b>	Une activité en pause ne reçoit pas les saisies de l'utilisateur et ne peut pas exécuter de code. Ce callback est appelée lorsque l'activité en cours est en pause et l'activité précédente a repris.
<b>onStop()</b>	Ce callback est appelé lorsque l'activité n'est plus visible.
<b>onDestroy()</b>	Ce callback est appelé avant que l'activité est détruite par le système.
<b>onRestart()</b>	Ce callback est appelée lorsque l'activité redémarre après un arrêt.



# Cycle de vie d'une activité

- Pour créer une activité, il suffit d'étendre la classe d'activité (ou une de ses sous classes et ré-implémenter les méthodes **onStop()**, **onStart()**, **onCreate()**, **onResume()**, **OnPause()**, **OnDestroy()**).
- Le code suivant donne le squelette typique pour une activité.



The screenshot shows the Android Studio interface. The left pane displays the project structure under 'UnExemple' with 'app' selected. Inside 'app', there are 'manifests', 'java', and 'res' directories. The 'java' directory contains a package 'com.example.unexample' which has a file 'MainActivity'. The right pane shows the code editor for 'MainActivity.java'. The code is as follows:

```
1 package com.example.unexample;
2
3 import ...
4
5
6 public class MainActivity extends AppCompatActivity {
7
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12     }
13
14 }
```



UnExemple app src main java com example unexemple MainActivity

1: Project

Build Variants

2: Structure

2: Favorites

Logcat

Layout Captures

TODO

Terminal

Build

Logcat

Profiler

Run

Event Log

Instant Run applied code changes and restarted the app. // (Don't show again) (a minute ago)

```
1 package com.example.unexemple;
2
3 import ...
4
5
6 public class MainActivity extends AppCompatActivity {
7     String msg="Cycle de vie: ";
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12         Log.i(msg, msg: "onCreate");
13     }
14
15     @Override
16     protected void onStart() {
17         super.onStart();
18         //L'activité va être visible
19         Log.i(msg, msg: "onStart");
20     }
21
22     @Override
23     protected void onResume() {
24         super.onResume();
25         Log.i(msg, msg: "onResume");
26     }
27
28 }
```

MainActivity &gt; onDestroy()



Gradle

Device File Explorer

# Conclusion

- Une application Android :
  - Série d'activités qui "vivent" et "meurent"
  - Une activité est généralement composée d'une interface graphique
  - Une activité est composée d'un contexte
- Une application ne limite pas le nombre d'activités mais le périphérique(émulateur par ex.) si.

# Evaluation

- Quels sont les différents types de fichiers dans un projet Android ?
- Remplissez le tableau suivant :

Callback	Evènement déclencheur de l'appel
<code>onCreate()</code>	
<code>onStart()</code>	
<code>onResume()</code>	
<code>onStop()</code>	
<code>onDelete()</code>	

# Webography

- <https://ionicframework.com/>
- <https://openclassrooms.com/fr/courses/5098931-developpez-une-application-mobile-multiplateforme-avec-ionic-3>
- <https://developer.xamarin.com/guides/>
- <https://angular.io/>
- <http://typescriptlang.org>
- <http://www.e-naxos.com/Blog/post/Strategie-de-developpement-Cross-Platform-Partie-2.aspx>
- Tutos Xamarin sur Google et Youtube entre autres.

# INF 3511 Programmation des Mobiles: Développement d'Applications Natives avec Android

**CRÉATION D'INTERFACES GRAPHIQUES**



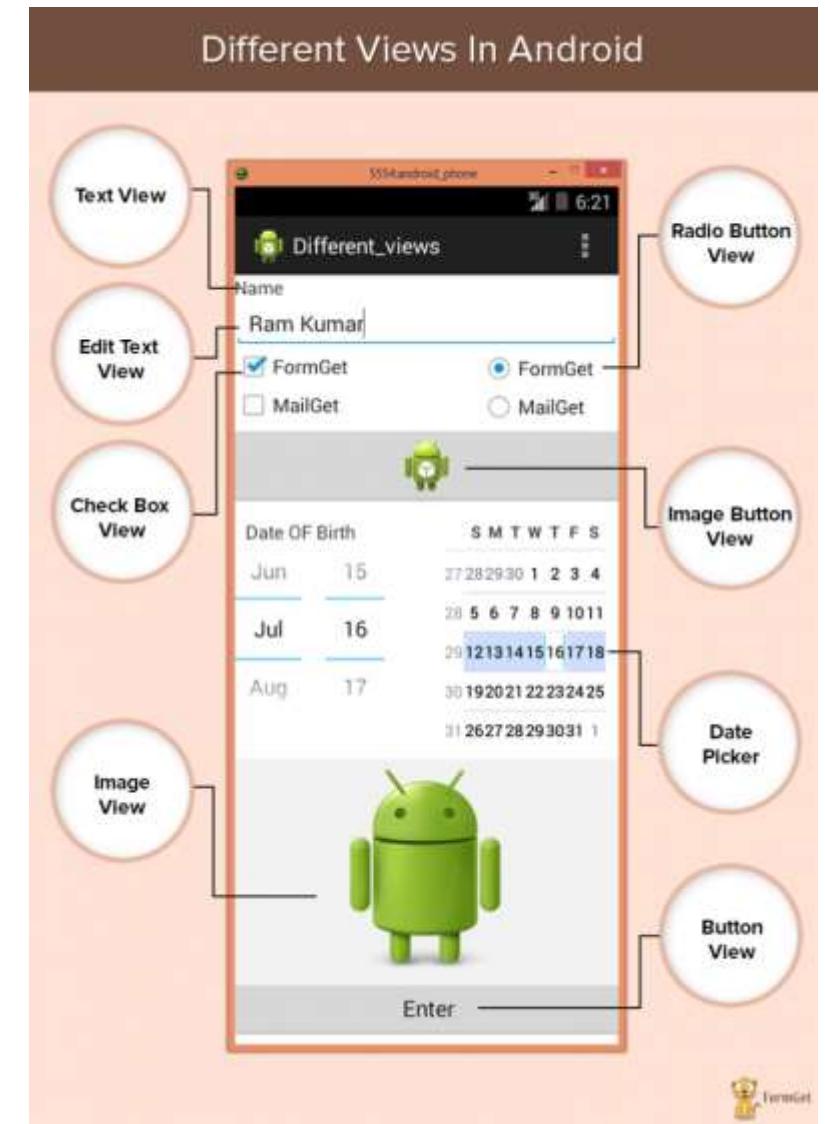
# Pensez vos interfaces pour un smartphone

- Ecran tactile de petite taille
  - Eviter les interfaces **trop touffues** (on ne peut pas agrandir l'écran comme on agrandit une fenêtre)
  - Eviter les éléments cliquables **trop petits** (il faut pouvoir cliquer avec le doigt même si on a des gros doigts)
  - Eviter les élément cliquables **trop tassés** (il faut pouvoir cliquer sur le bon élément même si on vise mal)
- Le défilement se fait par touché/glissé
  - **Pas trop d'ascenseurs** (on ne peut pas faire défiler un conteneur entier ET des éléments de ce conteneur dans le même sens)
  - Pas d'ascenseurs **mal placés** (si tous les éléments sont cliquables comment faire défiler sans cliquer ?)
- L'écran peut être tourné
- Tous les smartphones n'ont pas la même définition d'écran

# Création d'interfaces

- Par programme (comparable à java swing) mais avec des classes propres à Android
  - Définition de conteneurs (un conteneur = un conteneur + un mode de placement = JPanel + Layout)
  - Définition d'éléments d'interaction (vues) + placement et ajout dans les conteneurs
- Par description dans des fichiers xml (forme déclarative statique)
- Une interface est un arbre dont la racine est l'écran et les feuilles les éléments de l'interface (boutons, textes, cases à cocher, ...)

# Liste des vues Android sur l'interface de conception d'Android Studio



# Codage de l'interface utilisateur

- Un layout définit la structure visuelle pour une interface utilisateur, tels que l'interface utilisateur pour une activité ou widget. Il est de déclarer une disposition de deux manières:
  - Une déclaration en XML. Android fournit un vocabulaire XML simple qui correspond à une instance de la classe View ou de ses sous-classes. Sous Android Studio, les layouts sont déclarés dans les fichiers dans **res\layout**. Le fichier par défaut est **res\layout\activity\_main.xml**
  - Une instantiation des éléments du layout à l'exécution. Une application Android peut créer des objets de la classe View et ViewGroup (et manipuler leurs propriétés) par programmation. Sous Android Studio, cette option est gérée par dans les fichiers sources Java des activités (java/\*) : Il s'agit ici d'ajouter des vues par programmation Java

# Declaration XML: example de Activity main.xml

The screenshot shows the Android Studio interface with the project 'UnExemple' open. The code editor displays the XML file 'activity\_main.xml' containing the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/macky_sall"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Salut tout le monde !!!!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

A green callout box highlights the following text:

Ce fichier indique que cette application contient un seul widget (<TextView>) et l'organisation des vues est basée sur ConstraintLayout

The right side of the interface shows the 'Preview' tab, displaying a screenshot of a meeting with several people. Blue constraint lines are overlaid on the screen, illustrating how the TextView is positioned relative to the parent ConstraintLayout.

# Unités de mesure dans les fichiers XML

- Dans les fichiers XML, les dimensions des éléments d'interface (taille, marges, ...) peuvent être exprimées en diverses unités :
  - Pixels (**px**)
  - Pouces (**in**)
  - Millimètres (**mm**)
  - Points (**pt**) = 1/72 pouce
  - Pixel à densité indépendante (**dp**) 1 dp = 1 pixel pour un écran de 160 dpi
  - Pixel à taille indépendante (**sp**) relatif à la taille des polices de caractères
- Dans les fichiers XML les unités sont exprimées sous la forme : “24.5mm” ou “65px” ...

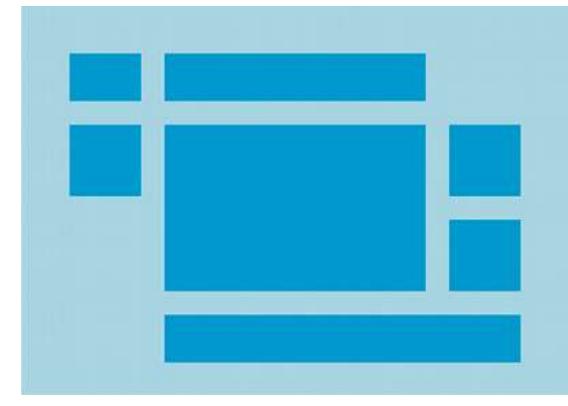
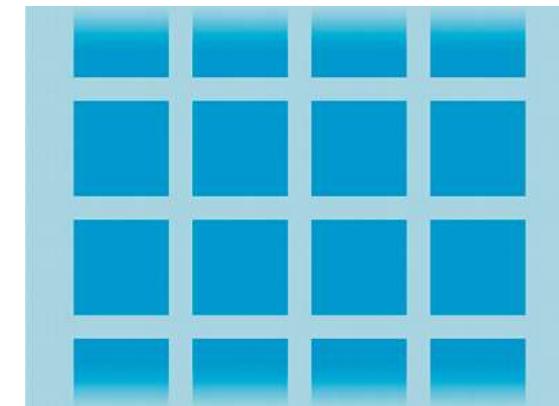
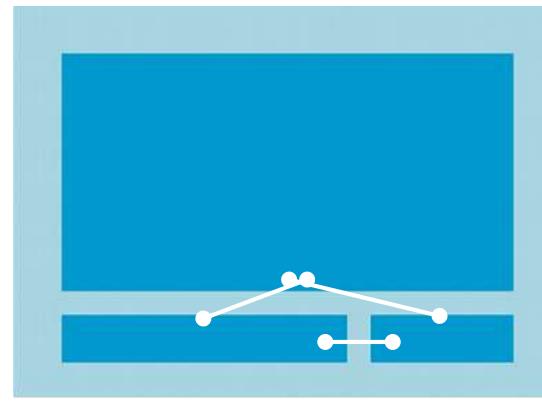
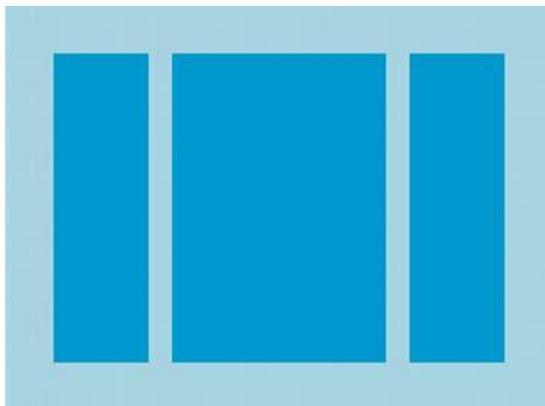
# Couleurs dans les fichiers XML

- Dans les fichiers XML, les couleurs sont exprimées sous la forme d'une chaîne de caractères codant les composantes en hexadécimal :  
**"#AARRVVBB"**
  - AA est l'opacité (00 totalement transparent, FF opaque)
  - RR est la composante rouge (00 à FF)
  - VV est la composante verte (00 à FF)
  - BB est la composante bleue (00 à FF)
- Si AA est omis la couleur est opaque

# ViewGroups pour layouts

- Les Layouts
  - sont des types spécifiques de ViewGroups (sous-classes de ViewGroup)
  - contiennent des vues d'enfants
  - peuvent être dans une rangée, colonne, grille, tableau, absolu

# Classes de mise en page communes



LinearLayout

ConstraintLayout

GridLayout

TableLayout

# Mise en page créée en XML

```
<LinearLayout  
    android:orientation="vertical"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <Button  
        ... />  
    <TextView  
        ... />  
    <Button  
        ... />  
</LinearLayout>
```

# Mise en page créée en Java Code de l'activité

```
LinearLayout linearL = new LinearLayout(this);
linearL.setOrientation(LinearLayout.VERTICAL);
TextView myText = new TextView(this);
myText.setText("Display this text!");
linearL.addView(myText);
setContentView(linearL);
```

# Définir la largeur et la hauteur en code Java

Set the width and height of a view:

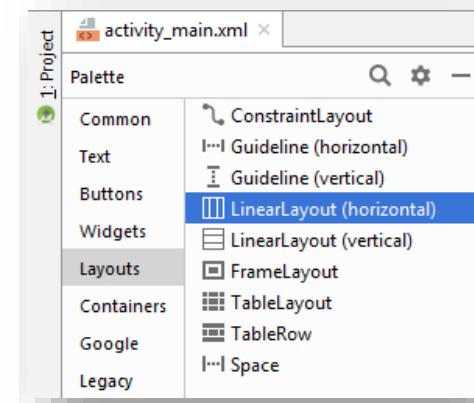
```
LinearLayout.LayoutParams layoutParams =  
    new LinearLayout.LayoutParams(  
        LayoutParams.MATCH_PARENT,  
        LayoutParams.MATCH_CONTENT);  
  
myView.setLayoutParams(layoutParams);
```

# Meilleures pratiques pour les hiérarchies de vues

- La hiérarchie des arrangements de vue affecte les performances de l'application
- Utilisez le plus petit nombre possible de vues les plus simples
- Maintenir la hiérarchie à plat - limiter l'imbrication des vues et des groupes de vues

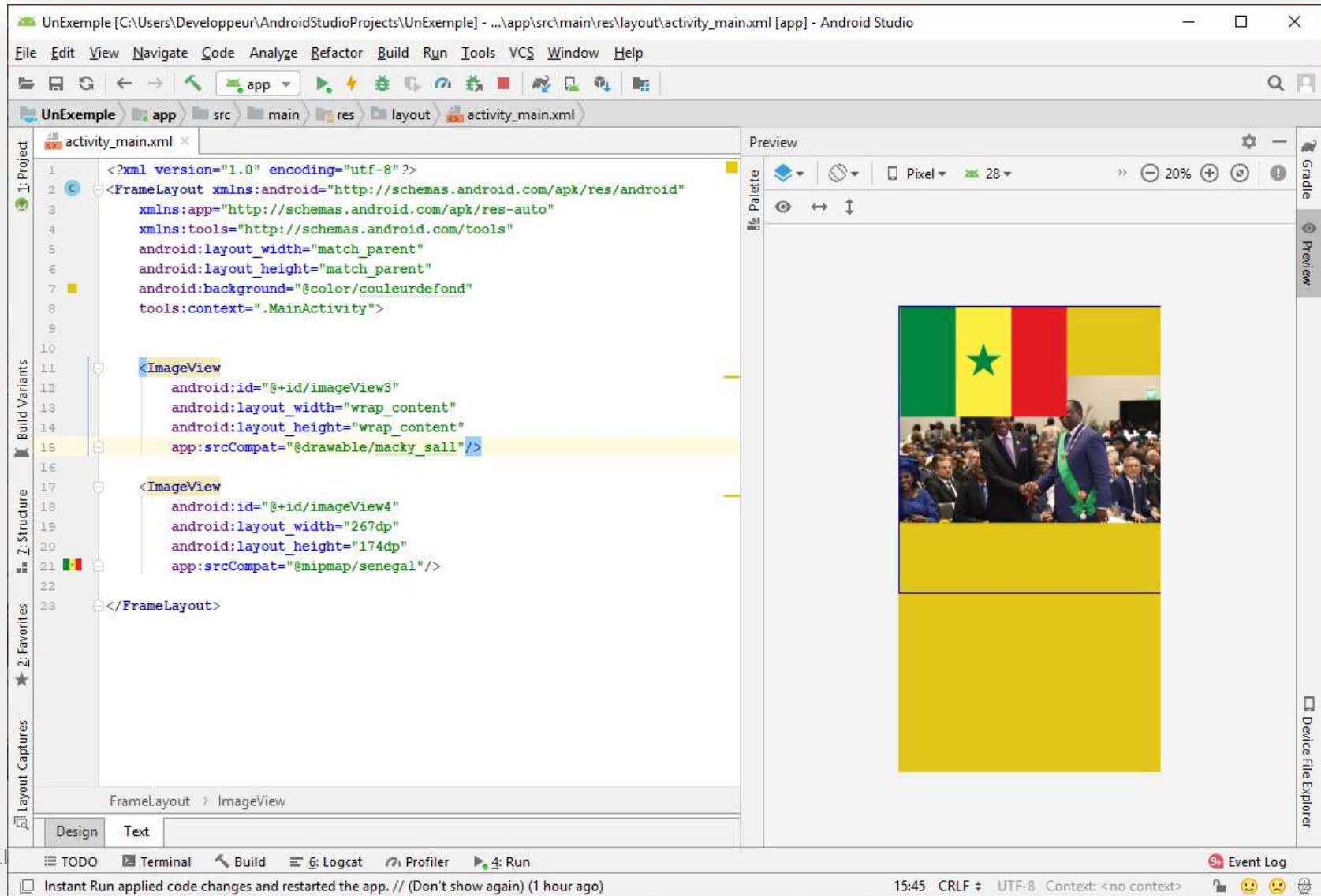
# Android Layout Managers: Les conteneurs

- Android offre différents conteneurs de mise en page pour permettre au développeur de créer n'importe quelle disposition avec une combinaison de ces dispositions de base:
  - **FrameLayout** (un seul élément)
  - **AbsoluteLayout** (plusieurs éléments placés par leur coordonnées): depracated
  - **LinearLayout** (plusieurs éléments placés horizontalement ou verticalement sans ascenseurs)
  - **TableLayout** (plusieurs éléments en tableau sans ascenseurs)
  - **RelativeLayout** (plusieurs éléments placés relativement les uns aux autres)
  - ...



# FrameLayout

Ne contient qu'un seul élément (si on en met plusieurs ils se superposent)  
Propriétés supplémentaires :  
*android:foreground* Pour définir une couleur ou une image.  
*android:foregroundGravity* Pour positionner l'image

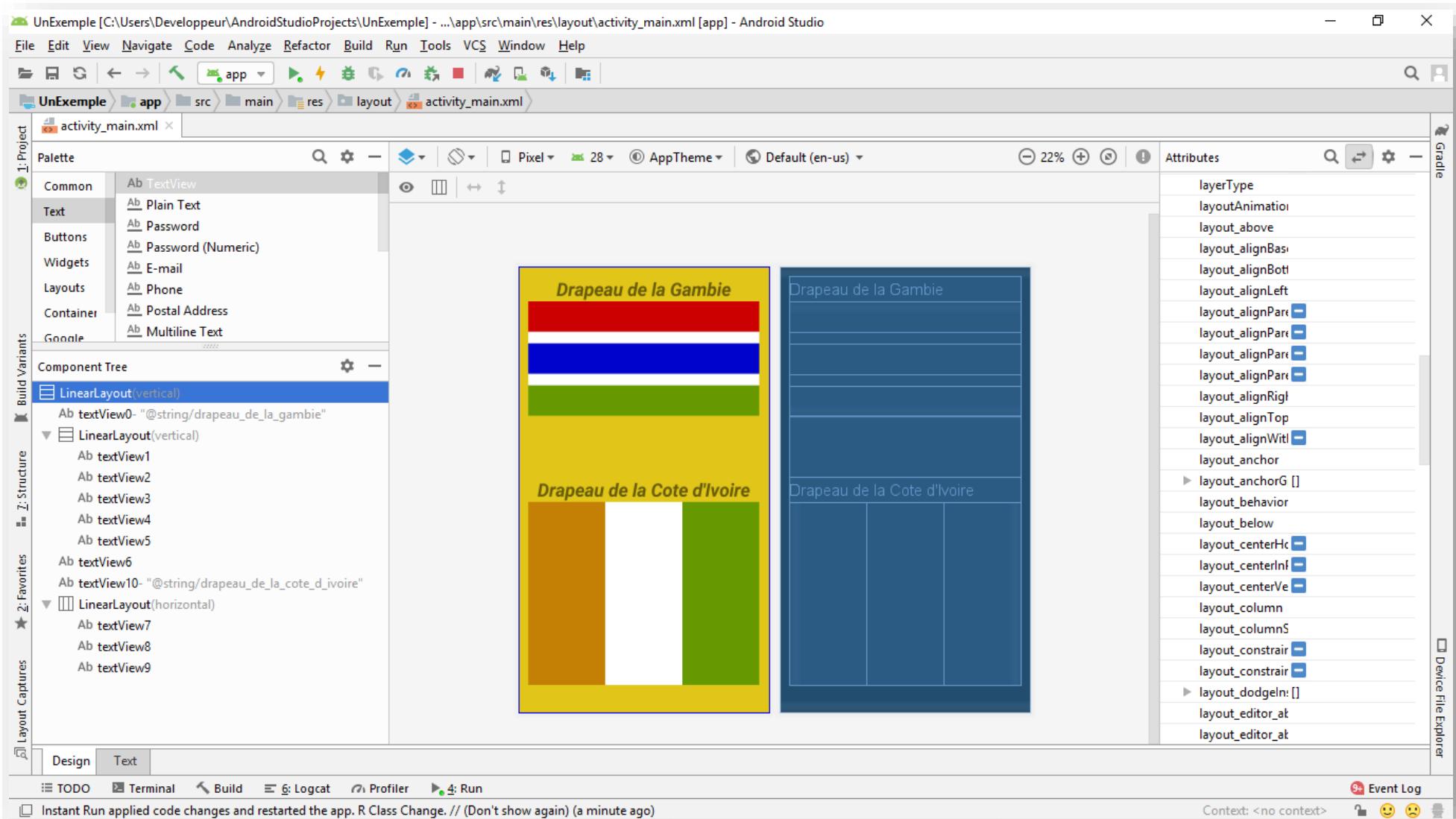


# LinearLayout

- **LinearLayout** est un *layout* de vues qui aligne tous les éléments à l'intérieur dans une seule direction, verticalement ou horizontalement. Il est possible de spécifier la direction de la mise en page avec l'attribut **android:orientation**.

Attribut	Description
<b>android:orientation</b>	Définit la direction dans laquelle les éléments vont s'étendre. Une valeur « horizontal » ou « vertical » les alignera verticalement ou horizontalement
<b>android:layout_width</b> <b>android:layout_height</b>	Définissent respectivement la largeur et la hauteur du layout. La valeur wrap_content signifie qu'il ne prendra que l'espace nécessaire pour afficher toutes les vues à l'intérieur. La valeur fill_parent va essayer de prendre toute l'espace disponible sur le parent.
<b>android:layout_weight</b>	Ce paramètre donne du poids au layout par rapport aux autres vues du parent. Plus la valeur est grande plus le layout courant prendra de l'espace au détriment des autres vues.

# LinearLayout pour les drapeaux Gambien et Ivoirien



# LinearLayout pour les drapeaux Gambiens et Ivoiriens

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:background="@color/couleurdefond"
8     android:orientation="vertical"
9     android:padding="16dp"
10    tools:context=".MainActivity">
11
12    <TextView
13        android:id="@+id/textView0"
14        android:layout_width="match_parent"
15        android:layout_height="wrap_content"
16        android:gravity="center"
17        android:text="@string/drapeau_de_la_gambie"
18        android:textSize="30sp"
19        android:textStyle="bold|italic" />
20
21    <LinearLayout
22        android:layout_width="match_parent"
23        android:layout_height="wrap_content"
24        android:orientation="vertical">
25
26        <TextView
27            android:id="@+id/textView1"
28            android:layout_width="match_parent"
29            android:layout_height="wrap_content"
30            android:height="50dp"
31            android:background="@android:color/holo_red_dark" />
32
33
34    <TextView
35        android:id="@+id/textView2"
36        android:layout_width="match_parent"
37        android:layout_height="wrap_content"
38        android:background="@android:color/background_light" />
39
40    <TextView
41        android:id="@+id/textView3"
42        android:layout_width="match_parent"
43        android:layout_height="wrap_content"
44        android:height="50dp"
45        android:background="#0003CC" />
46
47    <TextView
48        android:id="@+id/textView4"
49        android:layout_width="match_parent"
50        android:layout_height="wrap_content"
51        android:background="@android:color/background_light" />
52
53    <TextView
54        android:id="@+id/textView5"
55        android:layout_width="match_parent"
56        android:layout_height="wrap_content"
57        android:height="50dp"
58        android:background="@android:color/holo_green_dark" />
59
60
61    </LinearLayout>
```

# LinearLayout pour les drapeaux Gambiens et Ivoiriens

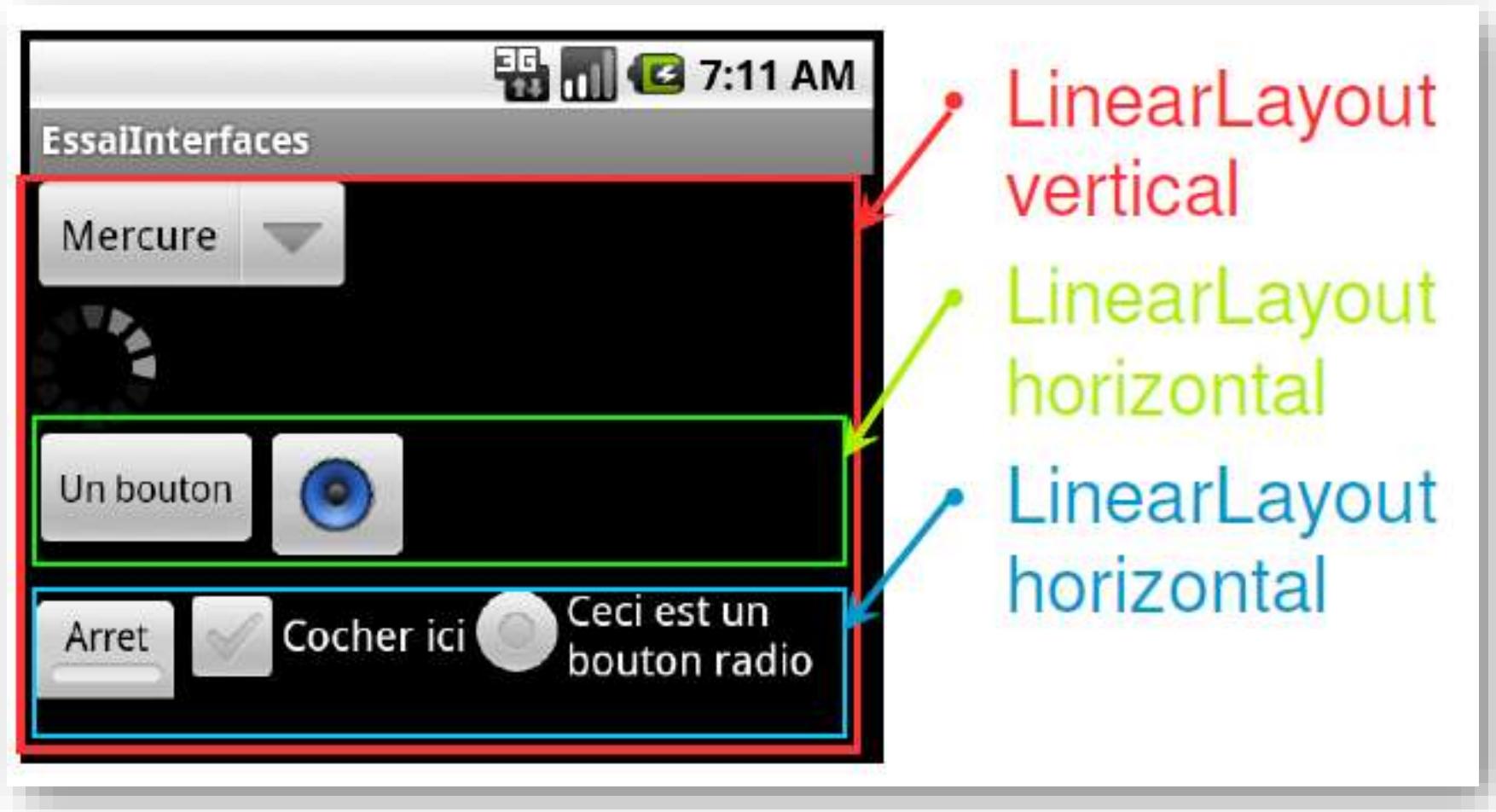


The screenshot shows an Android XML layout editor with two main sections. On the left, there is a code editor displaying the following XML code:

```
61 <TextView  
62     android:id="@+id/textView6"  
63     android:layout_width="match_parent"  
64     android:layout_height="wrap_content"  
65     android:height="100dp"/>  
66 <TextView  
67     android:id="@+id/textView10"  
68     android:layout_width="match_parent"  
69     android:layout_height="wrap_content"  
70     android:gravity="center"  
71     android:text="@string/drapeau_de_la_cote_d_ivoire"  
72     android:textSize="30sp"  
73     android:textStyle="bold|italic" />  
74 <LinearLayout  
75     android:layout_width="match_parent"  
76     android:layout_height="300dp"  
77     android:orientation="horizontal">  
78     <TextView  
79         android:id="@+id/textView7"  
80         android:layout_width="wrap_content"  
81         android:layout_height="match_parent"  
82         android:layout_weight="1"  
83         android:background="#C38005"/>  
84     <TextView  
85         android:id="@+id/textView8"  
86         android:layout_width="wrap_content"  
87         android:layout_height="match_parent"  
88         android:layout_weight="1"  
89         android:background="@android:color/background_light"/>  
90     </LinearLayout>  
91 </LinearLayout>
```

On the right, there is a preview window showing a horizontal layout with two text views. The first text view contains the text "drapeau de la côte d'ivoire" in bold italic font, centered. The second text view is below it, with its background color set to "#C38005".

# Exemple avec LinearLayout



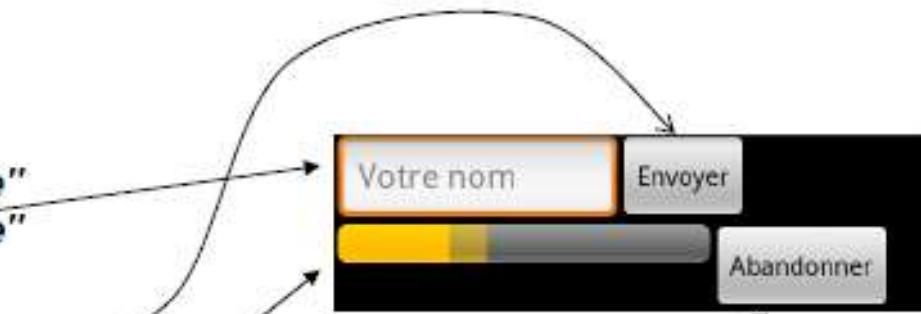
# RelativeLayout

- Sur cette méthode chaque élément est sensé définir sa position relativement aux autres éléments dans le même layout.
- Chaque élément défini sa position en fonction des éléments qui ont déjà été définies et les éléments suivants peuvent définir leur position en fonction du même élément. Les propriétés utilisées sont :

Propriété	Signification
<code>android:layout_alignParentTop</code>	Aligner sur le haut du layout
<code>android:layout_alignParentEnd</code>	Aligner à gauche du layout
<code>android:layout_alignParentStart</code>	Aligner à droite du parent
<code>android:layout_alignBottom</code>	Aligner en dessous du widget indiqué
<code>android:layout_toStartOf</code>	Aligner avant le widget indiqué
<code>android:layout_alignTop</code>	Aligner au-dessus du Widget indiqué

# Exemple avec RelativeLayout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_height="fill_parent"  
    android:layout_width="fill_parent">  
    <EditText android:id="@+id/nom"  
        ...  
        android:layout_alignParentTop="true"  
        android:layout_alignParentLeft="true"  
    />  
    <Button android:id="@+id/envoyer"  
        android:layout_toRightOf="@+id/nom"  
        ...  
    />  
    <ProgressBar android:id="@+id/attente"  
        android:layout_below="@+id/nom"  
        ...  
    />  
    <Button android:id="@+id/annuler"  
        android:layout_toRightOf="@+id/attente"  
        android:layout_below="@+id/nom"  
        ...  
    />  
</RelativeLayout>
```



# Exemple avec RelativeLayout

The screenshot shows an Android application interface. At the top, the title bar reads "Un Exemple". Below it is a yellow header bar containing a URL input field with the placeholder "URL: http://www.ugb.sn" and two buttons: "CANCEL" and "OK". The main content area is a black background with a white keyboard UI. The keyboard has a standard QWERTY layout with letters q, w, e, r, t, y, u, i, o, p in the first row, and a, s, d, f, g, h, j, k, l in the second row. Below these are rows for z, x, c, v, b, n, m, followed by a delete button (x) and a numeric/char row with ?123, , , ., and a backspace button. The bottom of the screen features a navigation bar with icons for back, forward, search, and other system functions.

Code Snippet from activity\_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/couleurdefond"
    android:padding="16dp"
    tools:context=".MainActivity">
    <TextView android:id="@+id/label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="URL:"
        android:paddingTop="15px"/>
    <EditText
        android:id="@+id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@+id/label"
        android:layout_alignBaseline="@+id/label"/>
    <Button
        android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/entry"
        android:layout_alignRight="@+id/entry"
        android:text="OK" />
    <Button
        android:id="@+id/cancel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@+id/ok"
        android:layout_alignTop="@+id/ok"
        android:text="Cancel" />
</RelativeLayout>
```

Project Structure:

- 1: Project
- Build Variants
- 2: Structure
- 3: Favorites
- Layout Captures

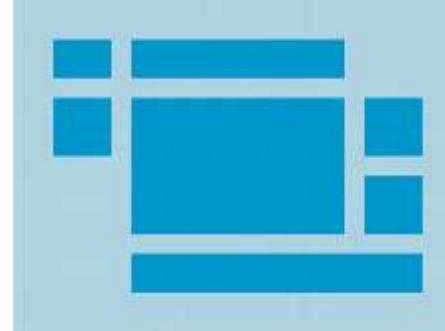
Bottom Navigation Bar:

- Design
- Text
- TODO
- Terminal
- Build
- Logcat
- Profiler
- Run
- Event Log

Bottom Status Bar:

- Gradle build finished in 12 s 361 ms (a minute ago)
- 8:27 CRLF: UTF-8 Context: <no context>
- Smiley faces: Happy, Neutral, Sad

# TableLayout



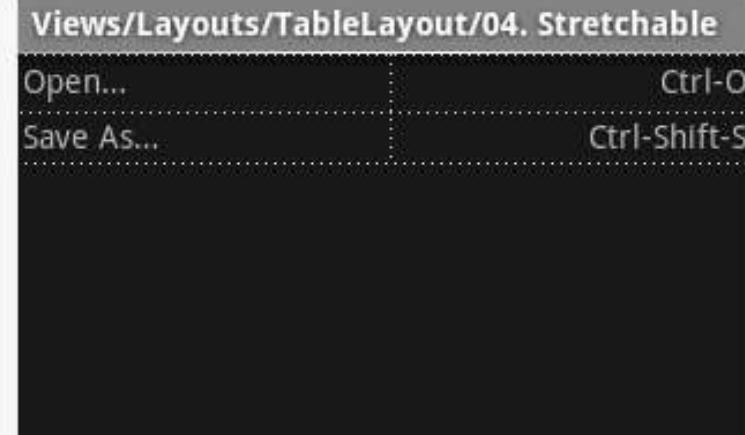
<https://developer.android.com/reference/android/widget/TableLayout.html>

- Pour placer des éléments en tableau sans ascenseurs (pour en avoir le mettre dans un ScrollView et/ou un HorizontalScrollView).
- Propriétés supplémentaires :
  - `android:collapseColumns` Pour définir les numéros de colonnes à cacher
  - `android:shrinkColumns` Pour définir les numéros de colonnes qui peuvent être rétrécies en fonction de la place disponible
  - `android:stretchColumns` Pour définir les numéros de colonnes qui peuvent être agrandies en fonction de leur contenu
  - Chaque élément ajouté dans un `TableLayout` indiquera le nombre de colonnes qu'il occupe en mettant dans ses propriétés : `android:layout_span` (par défaut 1)

# Exemple avec TableLayout

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="1">
    <TableRow>
        <TextView
            android:text="@string/table_layout_4_open"
            android:padding="3dip" />
        <TextView
            android:text="@string/table_layout_4_open_shortcut"
            android:gravity="right"
            android:padding="3dip" />
    </TableRow>

    <TableRow>
        <TextView
            android:text="@string/table_layout_4_save"
            android:padding="3dip" />
        <TextView
            android:text="@string/table_layout_4_save_shortcut"
            android:gravity="right"
            android:padding="3dip" />
    </TableRow>
</TableLayout>
```



# Exemple avec TableLayout

The screenshot shows the Android Studio interface with the following components:

- Project Bar:** Shows the project name "UnExemple" and the selected file "activity\_main.xml".
- Code Editor (Text Tab):** Displays the XML code for the TableLayout. The code defines a TableLayout with a single row containing an EditText and a View. The second row contains two buttons, "Cancel" and "OK".
- Preview Window:** Shows a preview of the layout in the "Design" tab. It features a yellow background with a white input field labeled "URL:" and two buttons at the bottom.
- Device Preview:** Shows the final UI on a virtual Android device. The screen has a green header with the title "Un Exemple". Below it is a yellow area with a white input field labeled "URL:". At the bottom are two buttons: "CANCEL" and "OK".
- Right Panel:** Contains various icons for navigating through the project and files.
- Bottom Bar:** Includes tabs for TODO, Terminal, Build, Logcat, Profiler, Run, Event Log, and a message about Instant Run.
- Page Footer:** Shows the page number "811".

# GridLayout



- La grille est composée d'un ensemble de lignes infiniment fines qui séparent la zone de visualisation en cellules. Dans l'API, les lignes de la grille sont référencées par des indices de la grille.
- Une grille avec  $N$  colonnes a  $N + 1$  indices de grille allant de 0 à  $N$  inclusivement.
- Quelle que soit la configuration de GridLayout, l'indice de grille 0 est fixé sur le bord d'attaque du conteneur et l'indice de grille  $N$  sur son bord de fuite (après la prise en compte du remplissage).

# GridLayout: exemple

The screenshot shows the Android Studio interface with the following components:

- Project Bar:** Shows the project structure: UnExemple > app > src > main > res > layout > activity\_main.xml.
- Code Editor:** Displays the XML code for `activity_main.xml`. The code defines a `GridLayout` with 3 columns and 4 rows, containing 5 buttons labeled "Button".

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/couleurdefond"
    android:columnCount="3"
    android:padding="16dp"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />

    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />

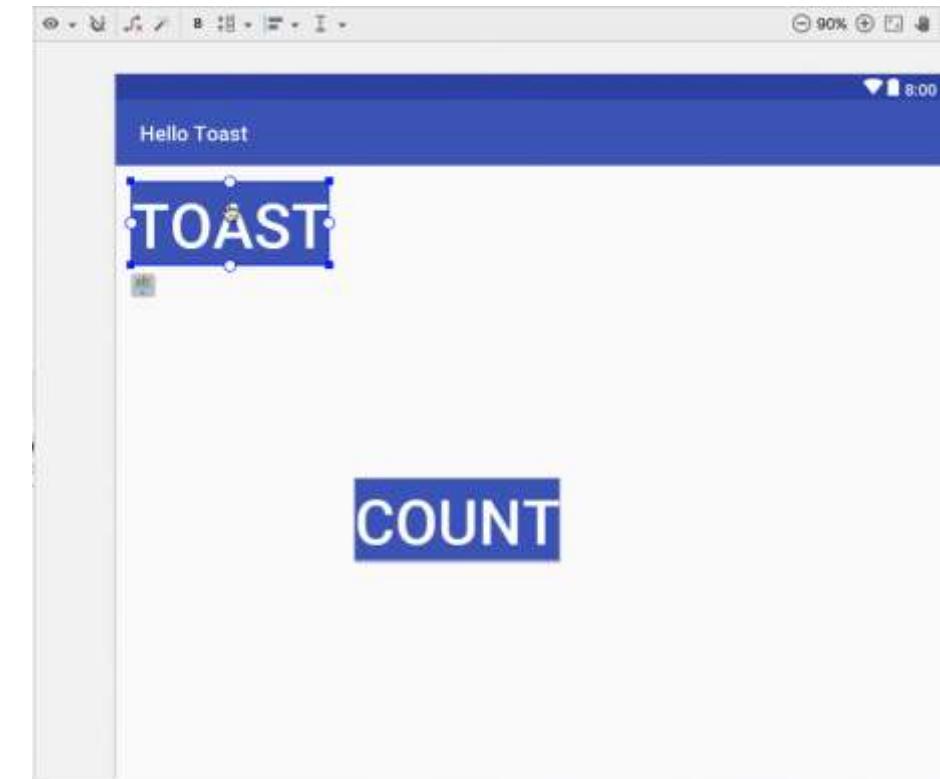
    <Button
        android:id="@+id/button4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />

    <Button
        android:id="@+id/button5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />
</GridLayout>
```
- Preview Panel:** Shows a preview of the layout in a mobile device window. The title bar says "Un Exemple". The layout consists of a 3x4 grid of buttons. The first three buttons in each row are visible, while the fourth button is partially visible at the bottom of each row.
- Device Preview:** Shows a full mobile device screen with the "Un Exemple" app running. The screen has a yellow background and a green header. The 3x4 grid of buttons is clearly visible.
- Bottom Bar:** Includes tabs for Design, Text, TODO, Terminal, Build, Logcat, Profiler, Run, Event Log, and Instant Run status.

# L'ÉDITEUR DE MISE EN PAGE ET CONSTRAINTLAYOUT

# L'éditeur de disposition avec ConstraintLayout

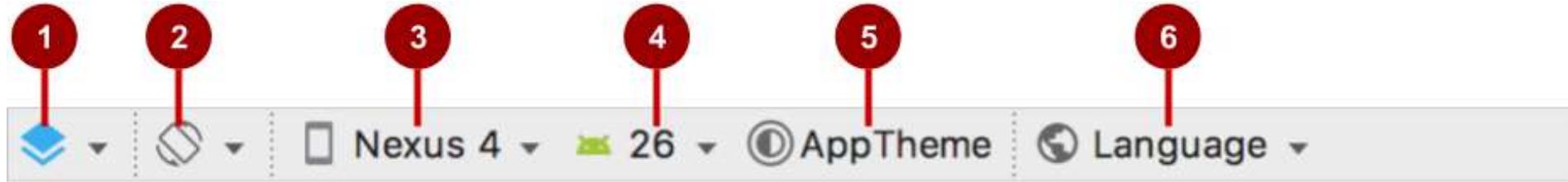
- Connecter des éléments d'interface utilisateur à la disposition parente
- Redimensionner et positionner des éléments
- Aligner des éléments sur les autres
- Ajuster les marges et les dimensions
- Changer les attributs



# Qu'est-ce que ConstraintLayout?

- Mise en page par défaut pour le nouveau projet Android Studio
- ViewGroup qui offre une flexibilité pour la conception de la disposition
- Fournit des contraintes pour déterminer les positions et l'alignement des éléments de l'interface utilisateur
- La contrainte est une connexion à une autre vue, à une mise en page parent ou à un repère invisible.

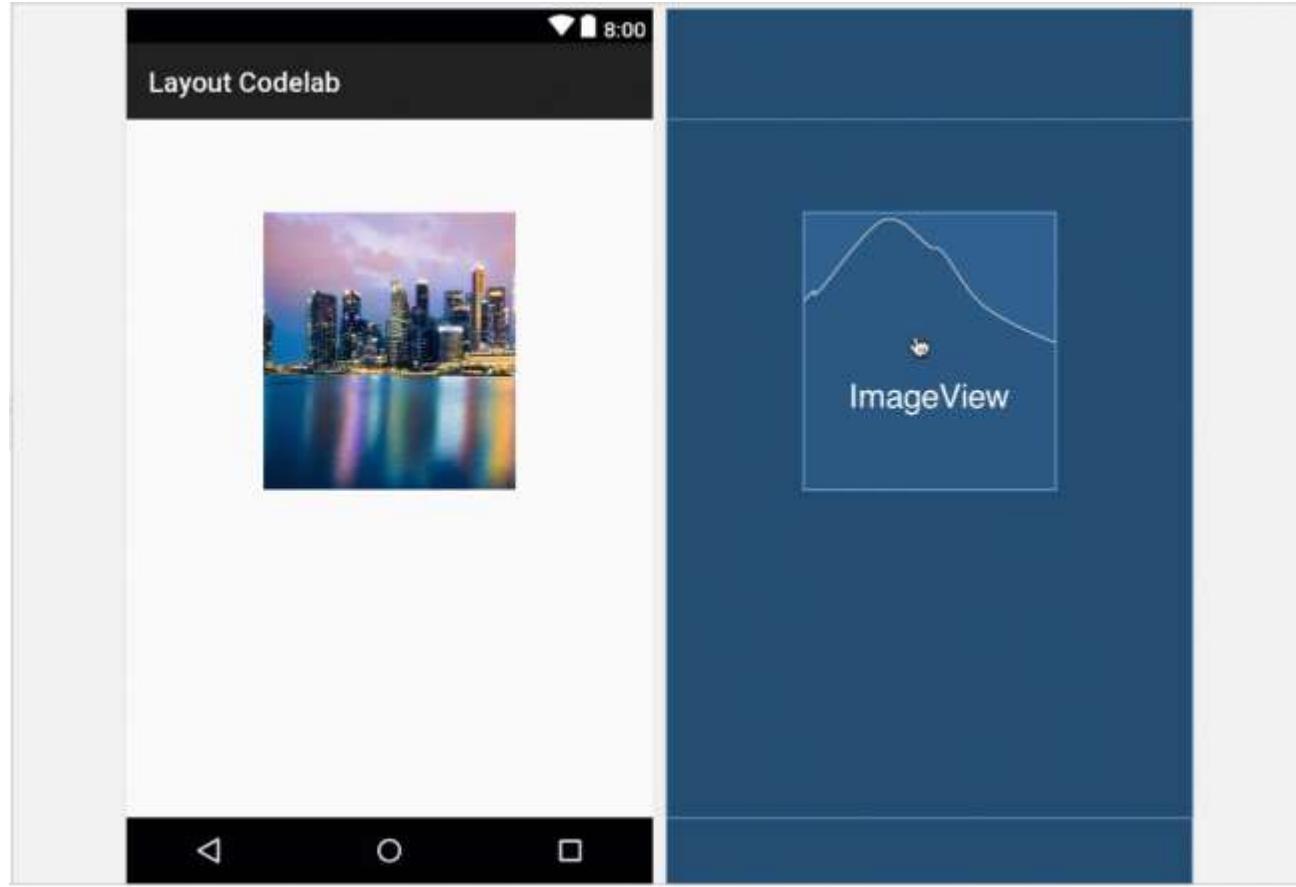
# Barre d'outils principale de l'éditeur de disposition



1. Sélectionner une surface de conception: vitres de conception et de plans
2. Orientation dans l'éditeur: Portrait et paysage
3. Périphérique dans l'éditeur: Choisissez le périphérique à prévisualiser
4. Version de l'API dans l'éditeur: Choisissez l'API pour l'aperçu
5. Thème dans l'éditeur: Choisissez un thème pour l'aperçu
6. Paramètres régionaux dans l'éditeur: choisissez la langue / les paramètres régionaux pour la prévisualisation

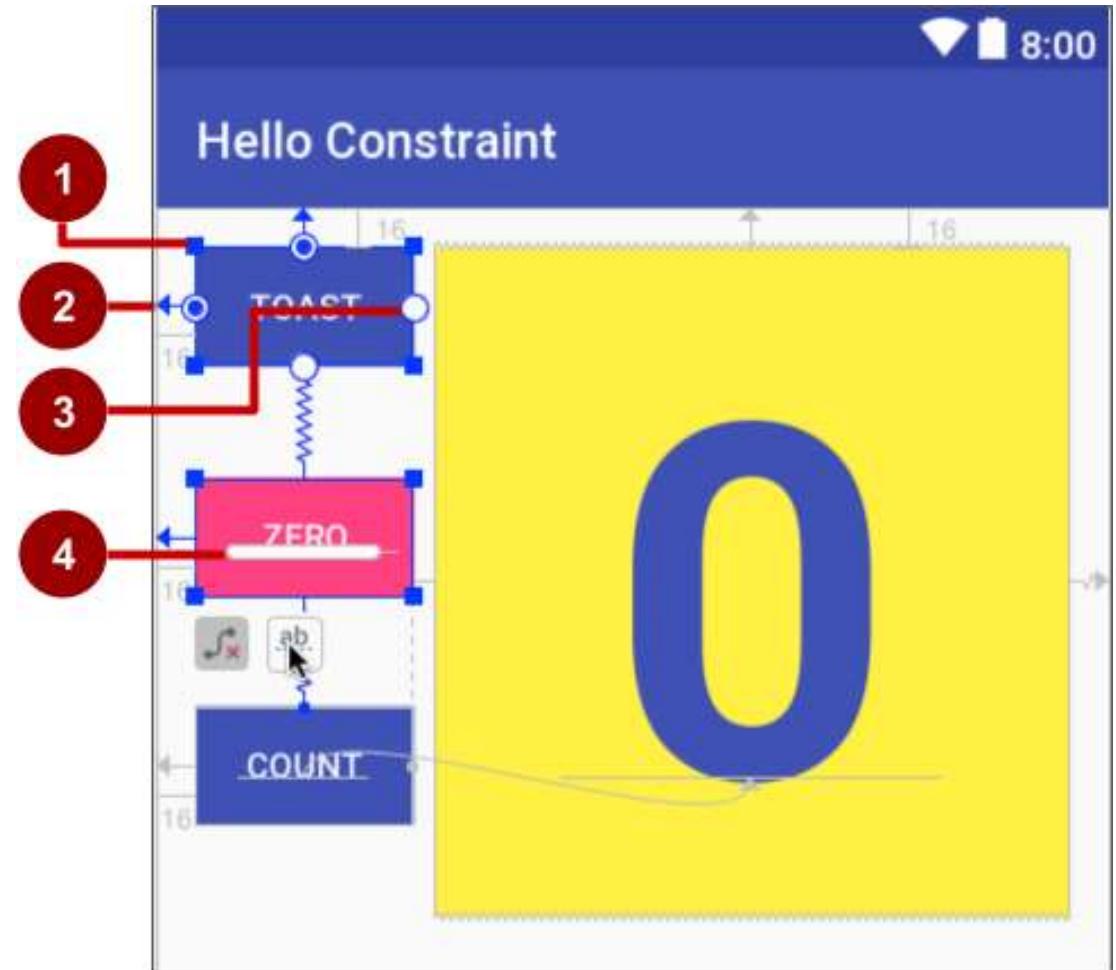
# Autoconnect

- Activer Autoconnect  dans la barre d'outils si elle est désactivée
- Faites glisser l'élément vers n'importe quelle partie d'une mise en page
- Autoconnect génère des contraintes par rapport à la mise en page parent



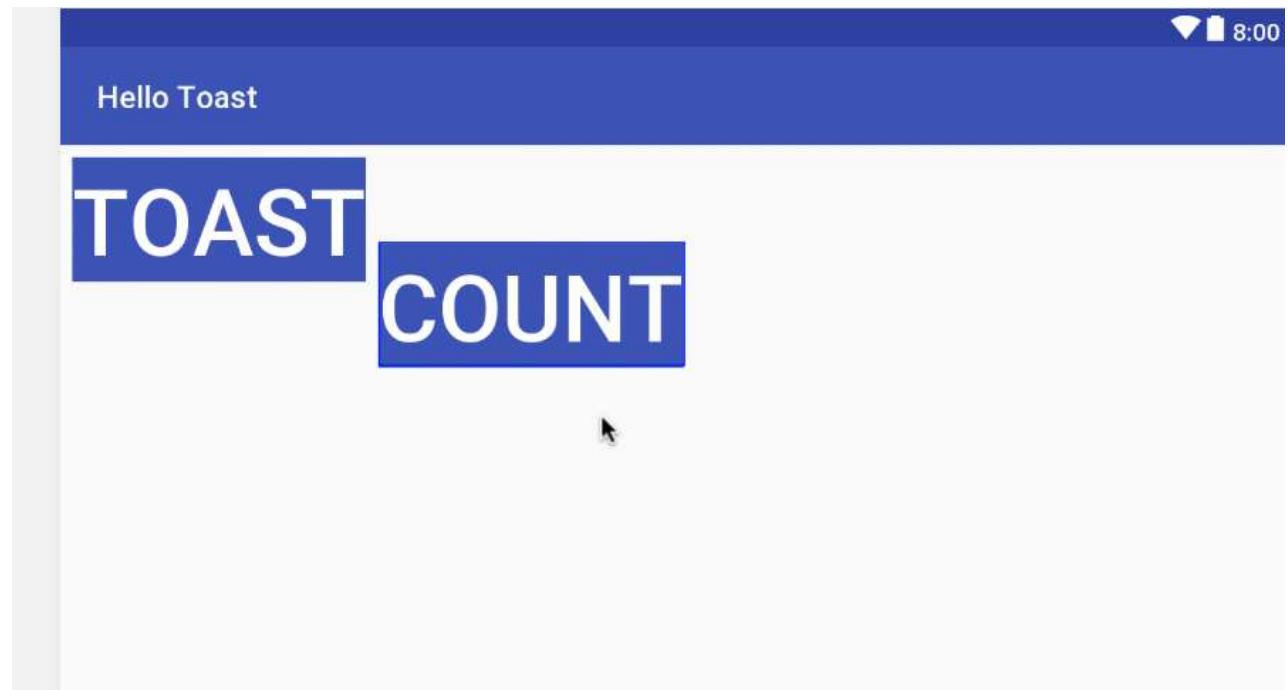
# Gestion des ConstraintLayout

1. Resizing handle
2. Constraint line and handle
3. Constraint handle
4. Baseline handle



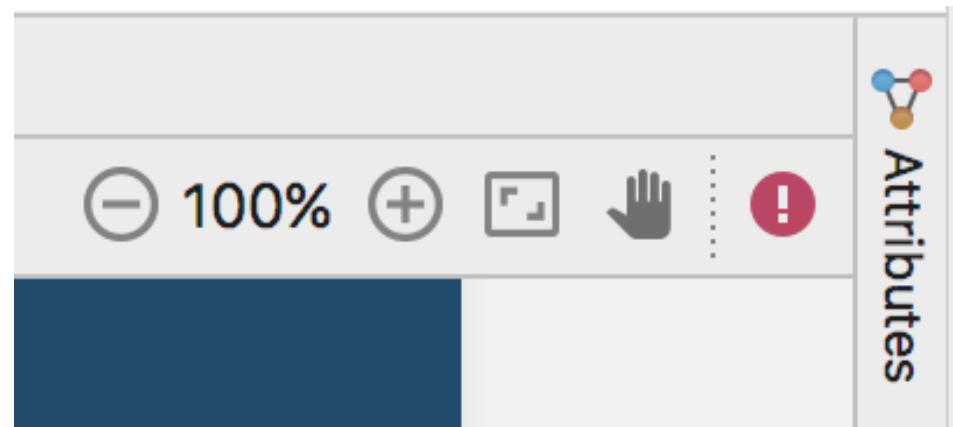
# Align elements by baseline

1. Cliquez sur  le bouton de contrainte de base
2. Glisser de la ligne de base vers la ligne de base d'un autre élément



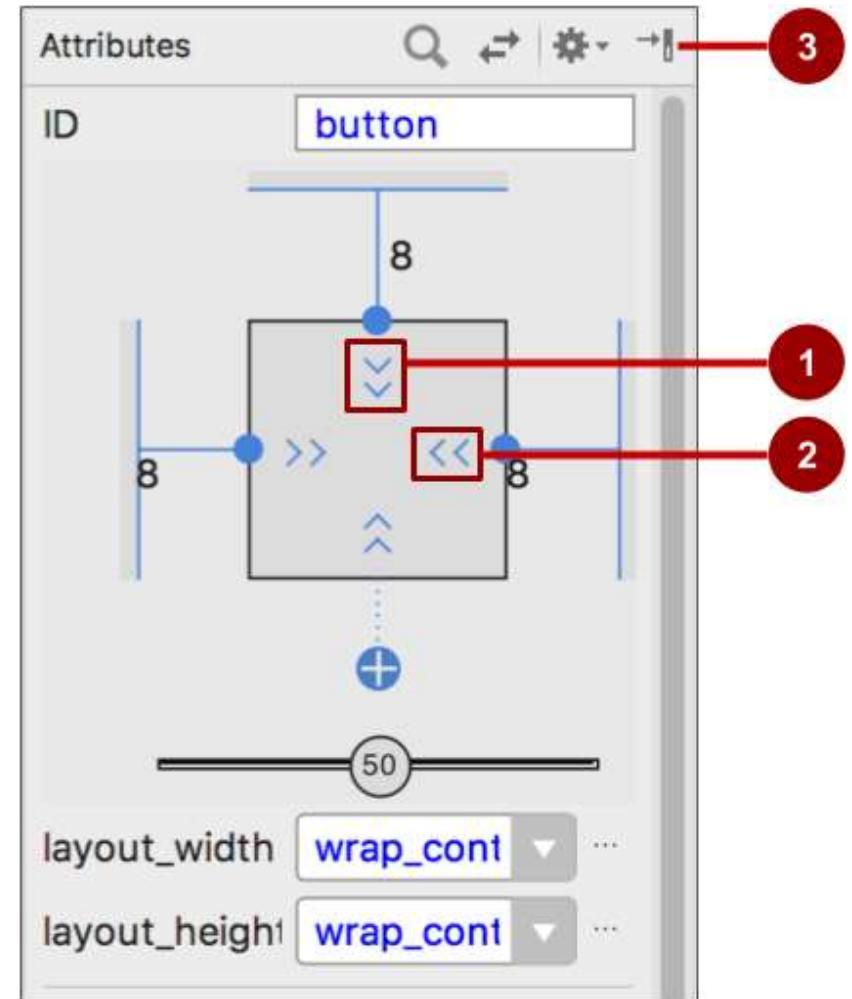
# Onglet des attributs

- Cliquez sur l'onglet Attributs
- L'Onglet des Attributs comprend:
  - Les Contrôles de marge pour le positionnement
  - Les Attributs tels que layout\_width



# Inspecteur de vue du volet Attributs

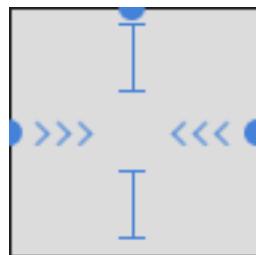
1. Le contrôle de la taille de la vue verticale spécifie `layout_height`
2. Le contrôle de la taille de la vue horizontale spécifie `layout_width`
3. Bouton de fermeture du volet Attributs



# Layout\_width et layout\_height

layout\_width et layout\_height changent avec les contrôles de taille

- ─ ┌─┐ match\_constraint: Développe l'élément pour remplir son parent
- ─ >>> wrap\_content: Réduit l'élément pour inclure du contenu
- ─ ─ Nombre fixe de dp (densité de pixels indépendants)...

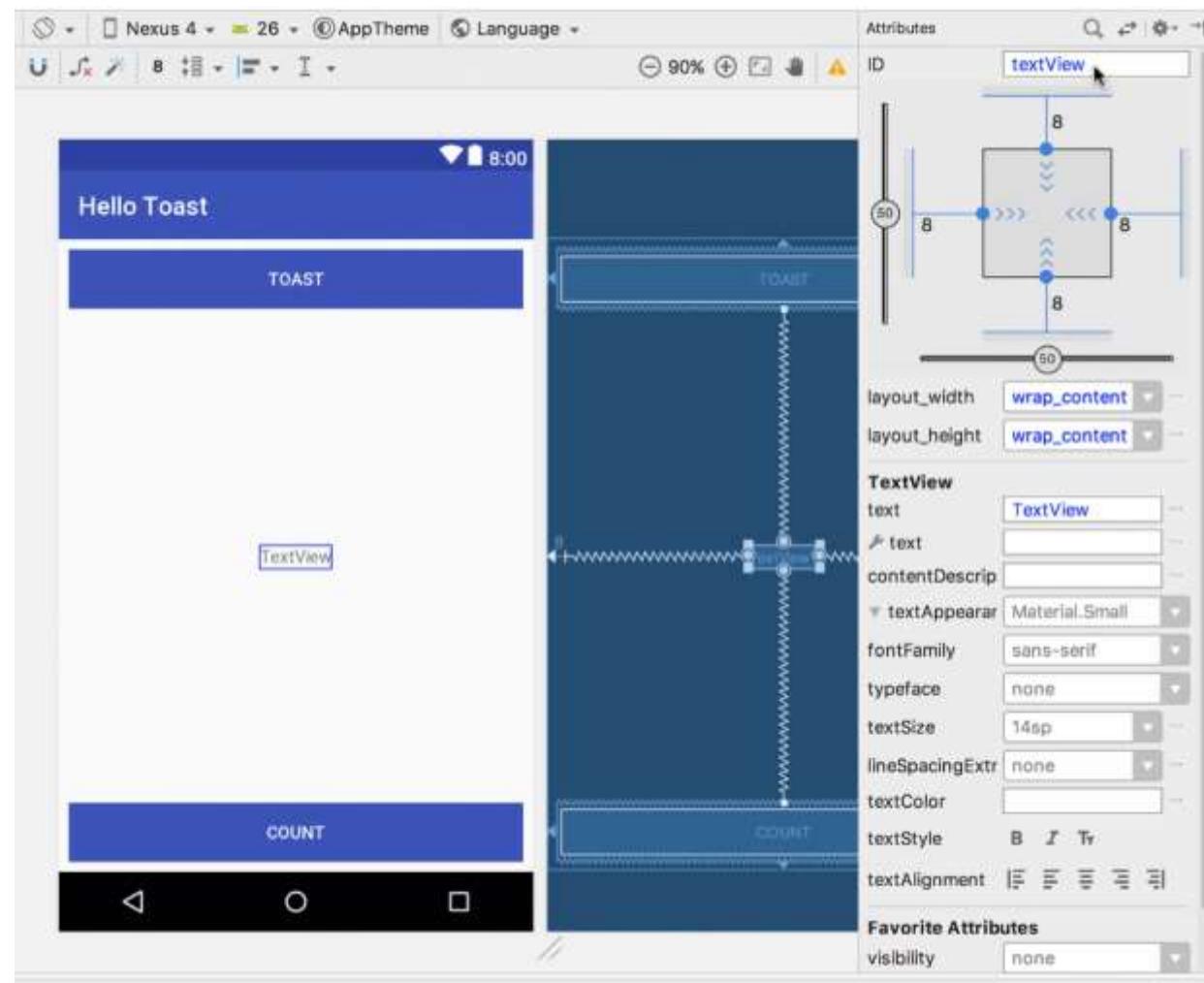


# Définir les attributs

Pour afficher et éditer tous les attributs d'un élément:

1. Cliquez sur l'onglet **Attributs**
2. Sélectionner un élément dans la conception, le plan directeur ou l'arborescence des composants
3. Changer les attributs les plus utilisés
4. Cliquez  en haut ou **Voir plus d'attributs** en bas pour voir et modifier plus d'attributs

# Exemple définition des attributs: TextView



# Prévisualisation des mises en page

Prévisualisation avec orientation horizontale / verticale:

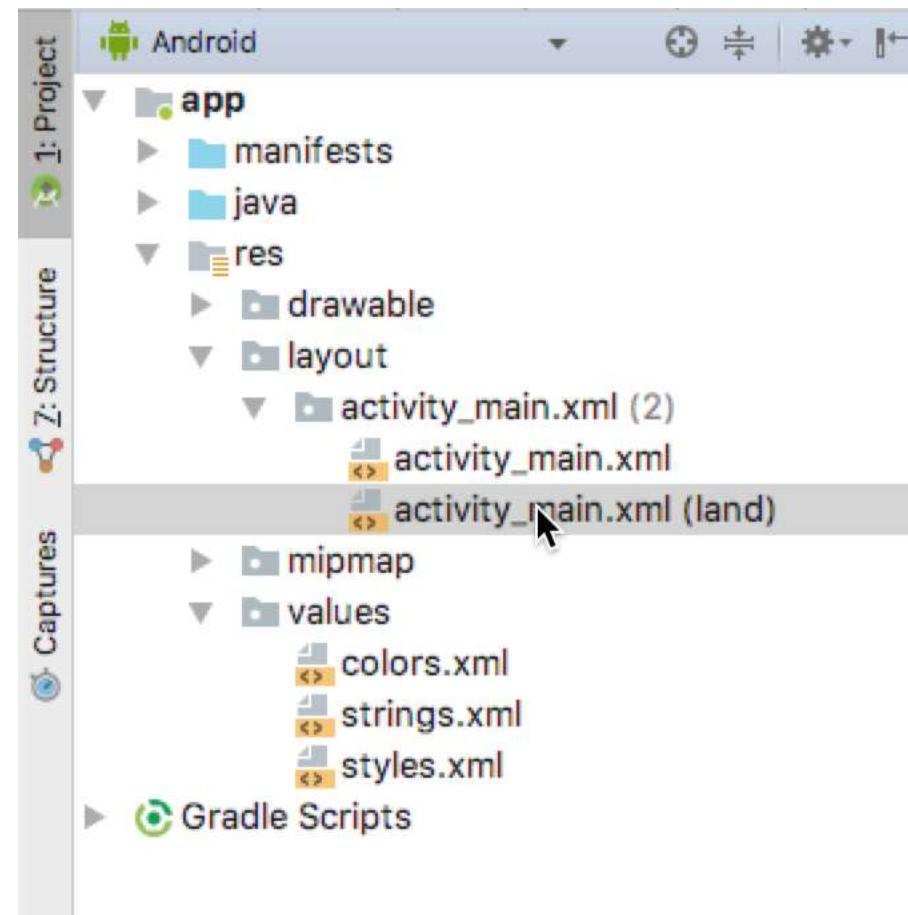
- Cliquez sur le bouton Orientation dans l'éditeur 
- Choisir **Switch to Landscape** ou **Switch to Portrait**

Aperçu de la mise en page avec différents appareils:

- Cliquez sur le bouton Appareil dans l'Editeur 
- Choisir l'appareil

# Créer une variante de mise en page pour paysage

1. Cliquez sur le bouton  Orientation dans l'éditeur
2. Choisissez **Create Landscape Variation**
3. Variante de modèle créée: `activity_main.xml (land)`
4. Editez la variante de présentation selon vos besoins



# Créer une variante de mise en page pour tablette

- Cliquez sur Orientation dans l'éditeur de mise en page. 
- Choisissez **Create layout x-large Variation**
- Variante de modèle créée: activity\_main.xml (xlarge)
- Editez la variante de présentation selon vos besoins

# Développement d'Applications Natives avec Android

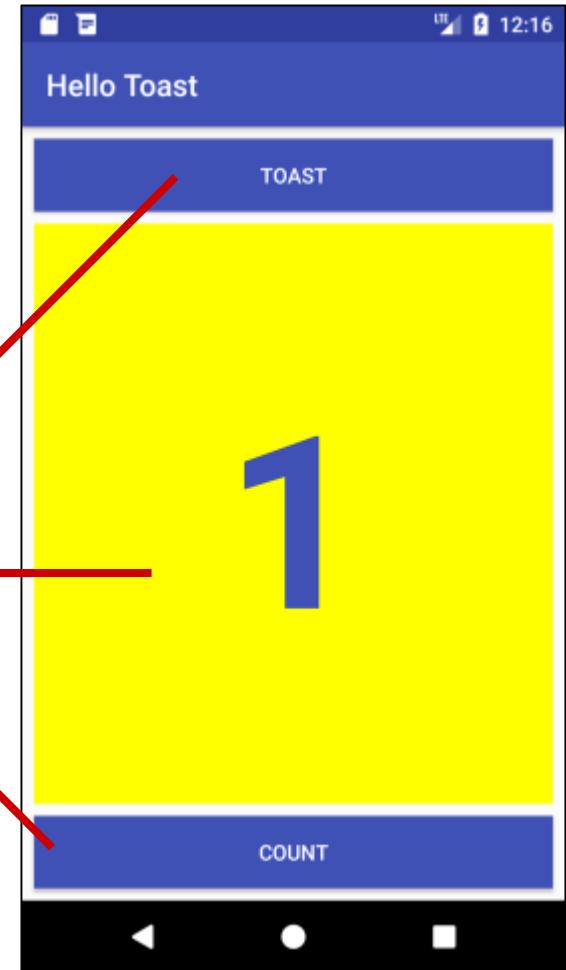
## ÉLÉMENTS D'INTERACTION: LES VUES



# Tout ce que VOUS voyez est une vue

Si vous regardez votre appareil mobile, chaque élément d'interface utilisateur que vous voyez est une **Vue(View)**.

Views

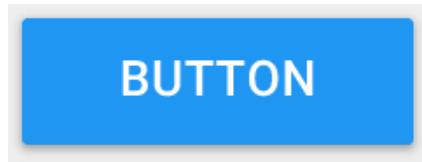


# Qu'est-ce qu'une vue ?

- Les sous-classes de vues sont des blocs de construction d'interface utilisateur de base
  - Afficher le texte (classe TextView), éditer le texte (classe EditText)
  - Boutons (classe de boutons), menus, autres contrôles
  - Scrollable (ScrollView, RecyclerView)
  - Afficher les images (ImageView)
  - Vues de groupe (ConstraintLayout et LinearLayout)

# Exemple de sous-classes de vues

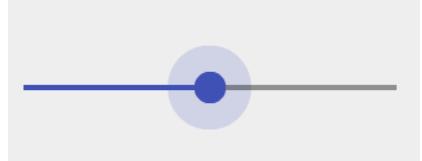
Button



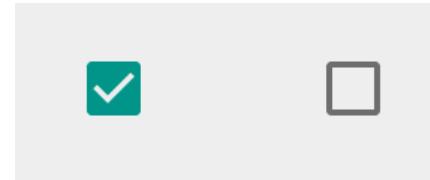
EditText



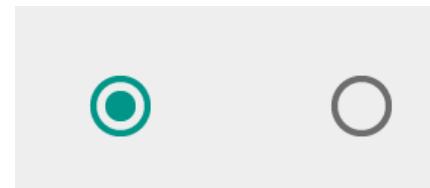
Slider



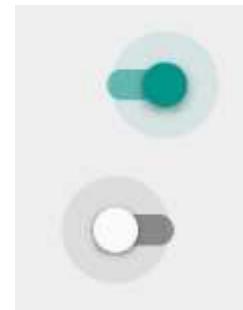
CheckBox



RadioButton



Switch



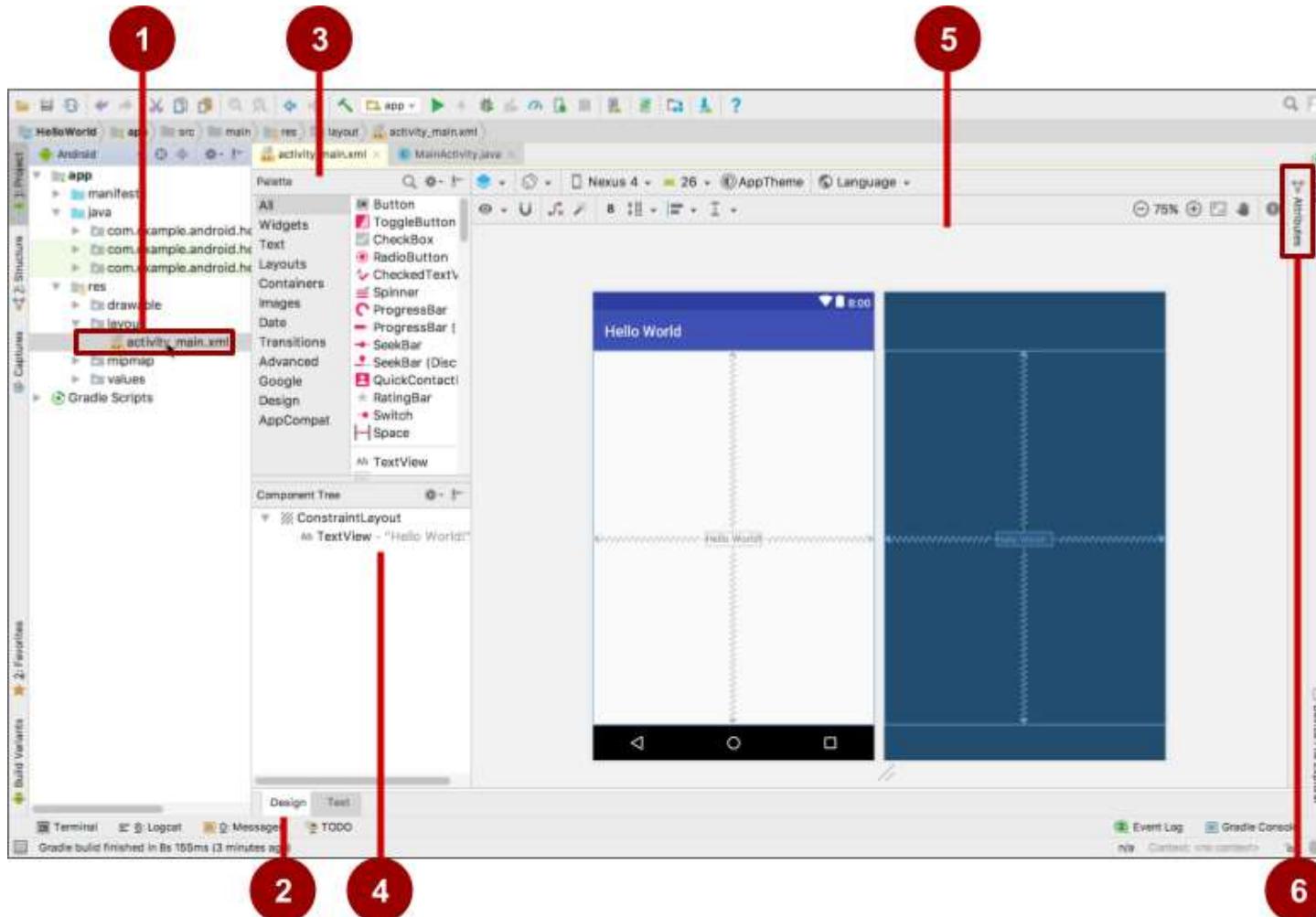
# Attributs des Vues

- Couleur, dimensions, positionnement(positioning)
- Peut avoir le focus (par exemple, sélectionné pour recevoir une entrée utilisateur)
- Peut être interactif (répondre aux clics de l'utilisateur)
- Peut être visible ou non
- Relations avec d'autres vues

# Créer des vues et des mises en page

- Editeur de disposition Android Studio: représentation visuelle de XML
- Éditeur XML
- Code Java

# Android Studio layout editor



1. Fichier de mise en page XML
2. Onglets Design et Codage des interfaces XML
3. Volet de la palette
4. Arbre des composants
5. Design et blueprint
6. Onglet Attributs

# Définir une vue sous XML

```
<TextView  
    android:id="@+id/show_count"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:background="@color/myBackgroundColor"  
    android:text="@string/count_initial_value"  
    android:textColor="@color/colorPrimary"  
    android:textSize="@dimen/count_text_size"  
    android:textStyle="bold" />
```

# Attributs des vues sous XML

**android:<property\_name>=<property\_value>**

**Example:** android:layout\_width="match\_parent"

**android:<property\_name>="@<resource\_type>/resource\_id"**

**Example:** android:text="@string/button\_label\_next"

**android:<property\_name>="@+id/view\_id"**

**Example:** android:id="@+id/show\_count"

# Créer une vue à partir du code Java

- In an Activity:

- `TextView myText = new TextView(this);`
- `myText.setText("Display this text!");`

*context*



# Quel est le context ?

- Context est une interface pour des informations globales sur un environnement d'application
- Obtenir le contexte: Contexte context = getApplicationContext();
- Une activité a son propre contexte: TextView myText = new TextView (this);

# Vues personnalisées

- Plus de 100 (!) Différents types de vues disponibles sur le système Android, tous les enfants de la classe View
- Si nécessaire, créez des vues personnalisées comme sous-classe des vues existantes ou la classe View.

# Principaux vues offerts par Android

- **Button**(<http://developer.android.com/reference/android/widget/Button.html>) : Un bouton cliquable.
- **CheckBox**(<http://developer.android.com/reference/android/widget/CheckBox.html>) : Une checkbox.
- **EditText**(<http://developer.android.com/reference/android/widget/EditText.html>) : Un champ de texte éditable.
- **DatePicker**(<http://developer.android.com/reference/android/widget/DatePicker.html>) : Sélection de dates.

# Principaux vues offerts par Android

- **RadioButton**(<http://developer.android.com/reference/android/widget/RadioButton.html>) : Représente les boutons radios.
- **Toast**(<http://developer.android.com/reference/android/widget/Toast.html>) : Un pop up message qui s'affiche sur l'écran.
- **ImageButton**(<http://developer.android.com/reference/android/widget/ImageButton.html>): Une image qui se comporte comme un bouton.
- ...

## Propriété communes aux éléments d'interface (conteneurs et vues)

- **Identifiant** : Un identifiant peut être associé à chaque élément décrit dans un fichier XML, cet identifiant permet d'accéder à l'objet créé dans le code ou de le référencer dans d'autres fichiers XML. Les éléments ne devant pas être référencés peuvent ne pas avoir d'identifiant.
- **android:id="@+id/mon\_ident"** permettra de retrouver cet élément par `findViewById(R.id.mon_ident)`.
- Méthode correspondante : **setId(int)**

# Propriété communes aux éléments d'interface (conteneurs et vues)

- **Visibilité**
  - `android:visibility` : Rend l'élément **visible**, **invisible** ou **absent** (avec **invisible** la place est conservée, avec **absent** la place n'est pas conservée .
- **Fond**
  - `android:background` couleur ou une image de fond
- **Taille**
  - `android:minHeight` et `android:minWidth` dimensions minimales
- **Placement des éléments contenus (défini pour chaque élément)**
  - `android:layout_height` et `android:layout_width` place prise par l'élément dans le conteneur :
    - **FILL\_PARENT** rempli toute la place
    - **WRAP\_CONTENT** occupe la place nécessaire
  - `android:layout_gravity` positionnement de l'élément dans le conteneur **top**, **bottom**, **left**, **right**, **center\_vertical**, **fill\_vertical**, **center\_horizontal**, **fill\_horizontal**, **center**, **fill**

# Les labels de texte avec XML (dans activity\_main.xml)

- TextView est normalement utilisé pour afficher un texte
- propriétés souvent utilisées:
  - **TextView**,
  - **EditText**,
  - **AutoCompleteTextView**,
  - **MultiCompleteTextView**.



```
<TextView  
    android:id="@+id/textView1"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="@string/hello_world" />
```

# Text-based vues: TextView

- Pour manipuler d'un TextView il faut récupérer la référence de l'objet avec la méthode `findViewById` de la classe Activity.
- La méthode **setText** de TextView permet de modifier le texte présenté tandis que la méthode **getText()** permet de récupérer le contenu actuelle du texte.

drawableStart

@android:drawable/ic\_menu\_save



UnExemple app src main res layout activity\_main.xml

activity\_main.xml x MainActivity.java

1: Project 1: Build Variants 1: Favorites 2: Favorites 1: Layout Captures

1 <?xml version="1.0" encoding="utf-8"?>  
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
3 xmlns:app="http://schemas.android.com/apk/res-auto"  
4 xmlns:tools="http://schemas.android.com/tools"  
5 android:layout\_width="match\_parent"  
6 android:layout\_height="match\_parent"  
7 android:background="@color/couleurdefond"  
8 android:orientation="vertical"  
9 android:padding="16dp"  
10 tools:context=".MainActivity">  
11  
12 <TextView  
13 android:id="@+id/textView0"  
14 android:layout\_width="match\_parent"  
15 android:layout\_height="match\_parent"  
16 android:background="#FF0000"  
17 android:gravity="center"  
18 android:text="@string/message"  
19 android:textSize="30sp"  
20 android:textStyle="bold|italic" />  
21  
22 </LinearLayout>  
23

Preview

Palette

23% 28 AppTheme Default (en-us)

Vous avez cliqué 0 fois sur ce TextView

Considérons l'application suivante qui donne un exemple d'utilisation du textView.

LinearLayout > TextView

Design Text

TODO Terminal Build Logcat Profiler Run

Gradle build finished in 21 s 531 ms (14 minutes ago) 18:34 CRLF UTF-8 Context: <no context>

Event Log

The screenshot shows the Android Studio interface with the project 'UnExemple' open. The left pane displays the Java code for `MainActivity.java`, which contains a click counter for a `TextView`. The right pane shows a virtual device running the application, displaying the text "Vous avez cliqué 7 fois sur ce TextView". A vertical toolbar on the right side of the screen provides quick access to various tools and settings.

```
1 package com.example.unexemple;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.util.Log;
6 import android.view.View;
7 import android.widget.TextView;
8
9 public class MainActivity extends AppCompatActivity {
10     TextView monTextView;
11     int compteur=1;
12     @Override
13     protected void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_main);
16         monTextView=(TextView)findViewById(R.id.textView0);
17         monTextView.setOnClickListener(new View.OnClickListener() {
18             @Override
19             public void onClick(View v) {
20                 monTextView.setText("Vous avez cliqué "+(compteur++)+" fois sur ce TextView");
21             }
22         });
23     }
24 }
```

The virtual device screen shows the application's title bar "Un Exemple" and the main content area containing the text "Vous avez cliqué 7 fois sur ce TextView".

# Les zones de texte : EditText

- Le widget EditText permet de donner des champs dans lesquelles l'utilisateur peut insérer du texte. Par exemple, l'application SMS utilise ce widget pour récupérer le contenu du message.
- Ce champ offre des fonctionnalités très intéressantes à travers l'attribut android:inputType. Cet attribut permet aussi bien de contrôler ce que l'utilisateur peut entrer ou bien la correction automatique des erreurs orthographiques.
- **Propriétés:**
  - android:autoText
  - android:capitalize
  - android:digits
  - android:singleLine
  - android:numeric
  - android:phoneNumber
  - android:password
  - android:inputType («number», «phone»,...)
  - android:hint

# Les zones de texte : EditText

- XML

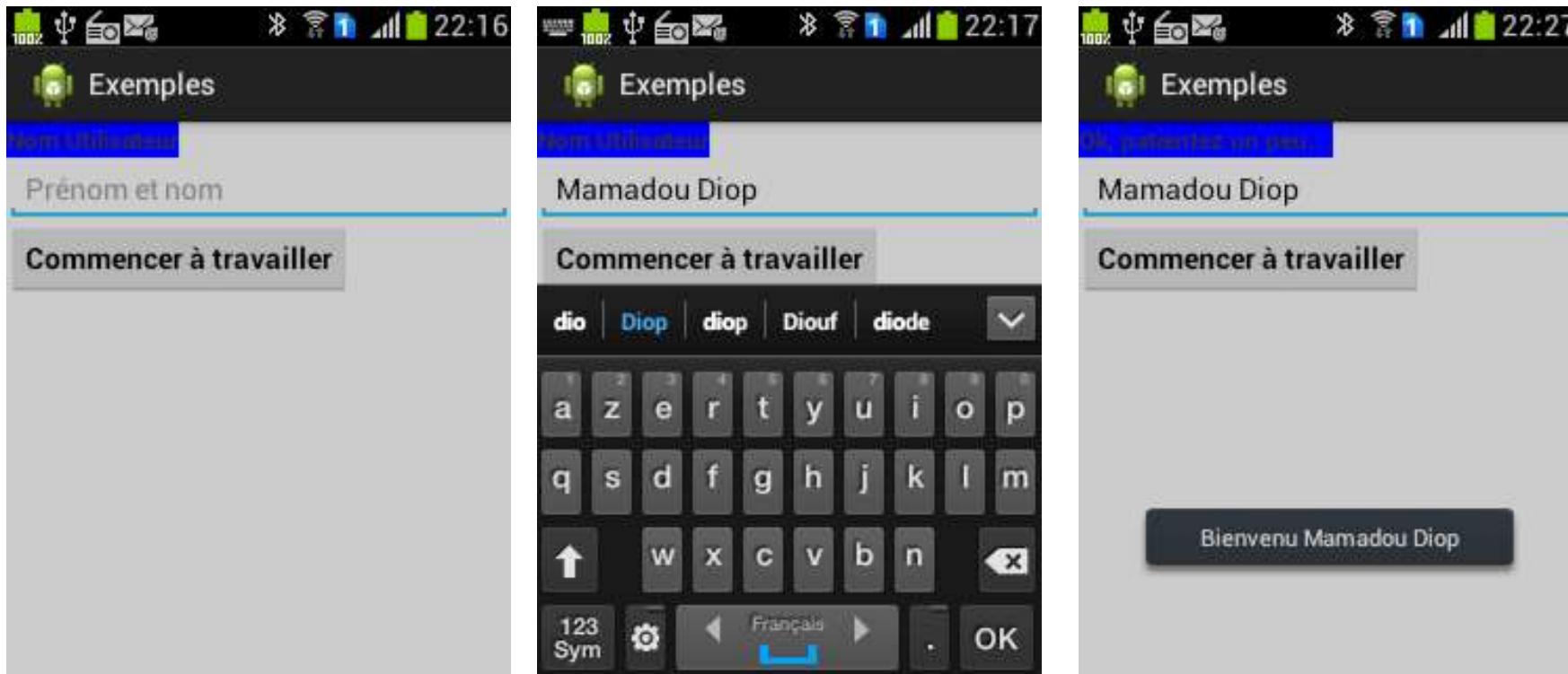
```
<EditText  
    android:id="@+id/EditText01"  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:text="">  
</EditText>
```

- Java

```
EditText edit = new EditText(this);  
edit.setText("Edite moi s'il te plaît ");  
myLayout.addView(edit);
```

# Zones de texte: exemple

Dans ce petit exemple, nous utiliserons un LinearLayout détenant un label (TexView), un textBox(EditText) et un bouton. Nous allons utiliser la vue comme une sorte d'écran de connexion simplifiée.



# Text-based vues: EditText

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffcccccc"
    android:orientation="vertical"
    tools:context=".MainActivity" >

    <TextView
        android:id="@+id/LabelUserName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#ff0000ff"
        android:text="Nom Utilisateur"
        android:textSize="16sp"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/txtUserName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:autoText="true"
        android:capitalize="words"
        android:ems="10"
        android:hint="Prénom et nom"
        android:inputType="textPersonName"
        android:textSize="18sp" />

    <Button
        android:id="@+id/bouttonCommencer"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Commencer à travailler"
        android:textSize="14px"
        android:textStyle="bold" />
</LinearLayout>
```

# Text-based view: EditText

```
1 package com.example.exemples;
2
3*import android.os.Bundle;*
4
5 public class MainActivity extends Activity implements OnClickListener {
6     Button monButton;
7     EditText txtUserName;
8     TextView labelUserName;
9
10    @Override
11    protected void onCreate(Bundle savedInstanceState) {
12        super.onCreate(savedInstanceState);
13        setContentView(R.layout.activity_main);
14        labelUserName = (TextView) findViewById(R.id.LabelUserName);
15        txtUserName = (EditText) findViewById(R.id.txtUserName);
16        monButton = (Button) findViewById(R.id.buttonCommencer);
17        monButton.setOnClickListener(this);
18    }
19
20    @Override
21    public void onClick(View v) {
22        // TODO Auto-generated method stub
23        if (v.getId() == monButton.getId()) {
24            String userName = txtUserName.getText().toString();
25            if (userName.compareTo("Mamadou Diop") == 0) {
26                labelUserName.setText("Ok, patientez un peu...");
27                Toast.makeText(getApplicationContext(), "Bienvenu " + userName,
28                    Toast.LENGTH_LONG).show();
29            }
30            Toast.makeText(getApplicationContext(), "Bienvenu " + userName,
31                    Toast.LENGTH_LONG).show();
32        }
33    }
34}
```

# Button

- Les boutons sont des contrôles qui communiquent les événements lorsque l'utilisateur les touche avec le tactile.
- Un bouton peut contenir du texte, une image ou les deux.
- L'attribut **android:text** permet d'ajouter du texte sur l'image alors que les attributs **android:drawable\*** permettent d'insérer une icône et sur le bouton.

# Les boutons : Button

- Button : Mêmes paramètres que TextView
- ImageButton : Mêmes paramètres que ImageView càd :
  - android:src="couleur" pour définir une couleur ou une image
  - android:adjustViewBounds Pour indiquer si la taille du bouton doit ou pas être ajustée à celle de l'image
  - android:baselineAlignBottom Pour indiquer que l'image est placée ou pas en bas de la zone
  - android:cropToPadding Pour indiquer si l'image sera coupée ou pas si elle est plus grande que la taille disponible
  - android:scaleType="s" (où s peut prendre les valeurs : matrix, fitXY, fitStart, fitCenter, fitEnd, center, centerCrop, centerInside) permet de redimensionner ou pas l'image à la taille disponible et/ou de la déformer.
  - android:maxHeight Pour définir la hauteur disponible
  - android:maxWidth Pour définir la largeur disponible
  - android:tint Pour définir une couleur qui teinte l'image

# Les boutons

```
<Button  
    android:id="@+id/Button01"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Clique sur moi " >  
</Button>
```

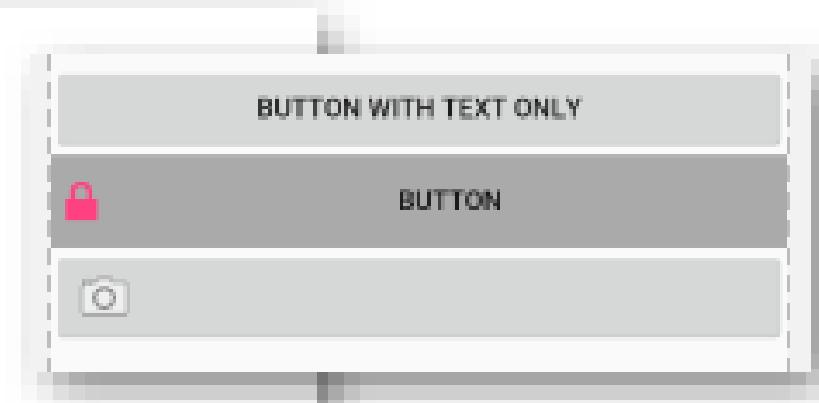
```
Button b = (Button) findViewById(R.id.Button01);  
b.setOnClickListener(new OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        Toast.makeText(v.getContext(), "Stop ", Toast.LENGTH_LONG).show();  
    }  
});  
myLayout.addView(b);
```

# Button

```
<Button  
    android:text="Button with text only"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/button4" />
```

```
<Button  
    android:text="Button"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/button5"  
    android:drawableLeft="@android:drawable/ic_lock_lock"  
    android:background="@android:color/darker_gray"  
    tools:ignore="RtlHardcoded" />
```

```
<Button  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/button6"  
    android:drawableLeft="@android:drawable/ic_menu_camera" />
```



# ImageButton

- Bouton normal, mais avec une image. Mêmes paramètres XML que ImageView
- Exemple de fichier XML :

```
<ImageButton android:id="@+id/boutonloupe"
    android:src="@drawable/macky_sall"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```
- **Méthodes de la classe ImageButton**
  - android.widget.ImageButton
  - ImageButton hérite de ImageView il a donc les mêmes sauf le constructeur :
- ***Construction:*** ImageButton(Context) le paramètre est généralement l'activité elle-même

UnExemple > app > src > main > res > layout > activity\_main.xml

activity\_main.xml > MainActivity.java

Preview

16% 16% 16%

Gradle Preview

1: Project

2: Favorites

3: I: Structure

4: Favorites

5: Layout Captures

6: TODO Terminal Build Logcat Profiler Run

Instant Run applied code changes and restarted the current activity. // (Don't show again) (moments ago)

18:50 CRLF UTF-8 Context: <no context>

Event Log

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/couleurdefond"
    android:orientation="vertical"
    android:padding="16dp"
    tools:context=".MainActivity">

    <ImageButton
        android:id="@+id/imageMacky"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:contentDescription="@string/todo"
        android:src="@drawable/macky_sall"
        android:layout_gravity="center_horizontal"/>

</LinearLayout>
```

LinearLayout > ImageButton

Design Text

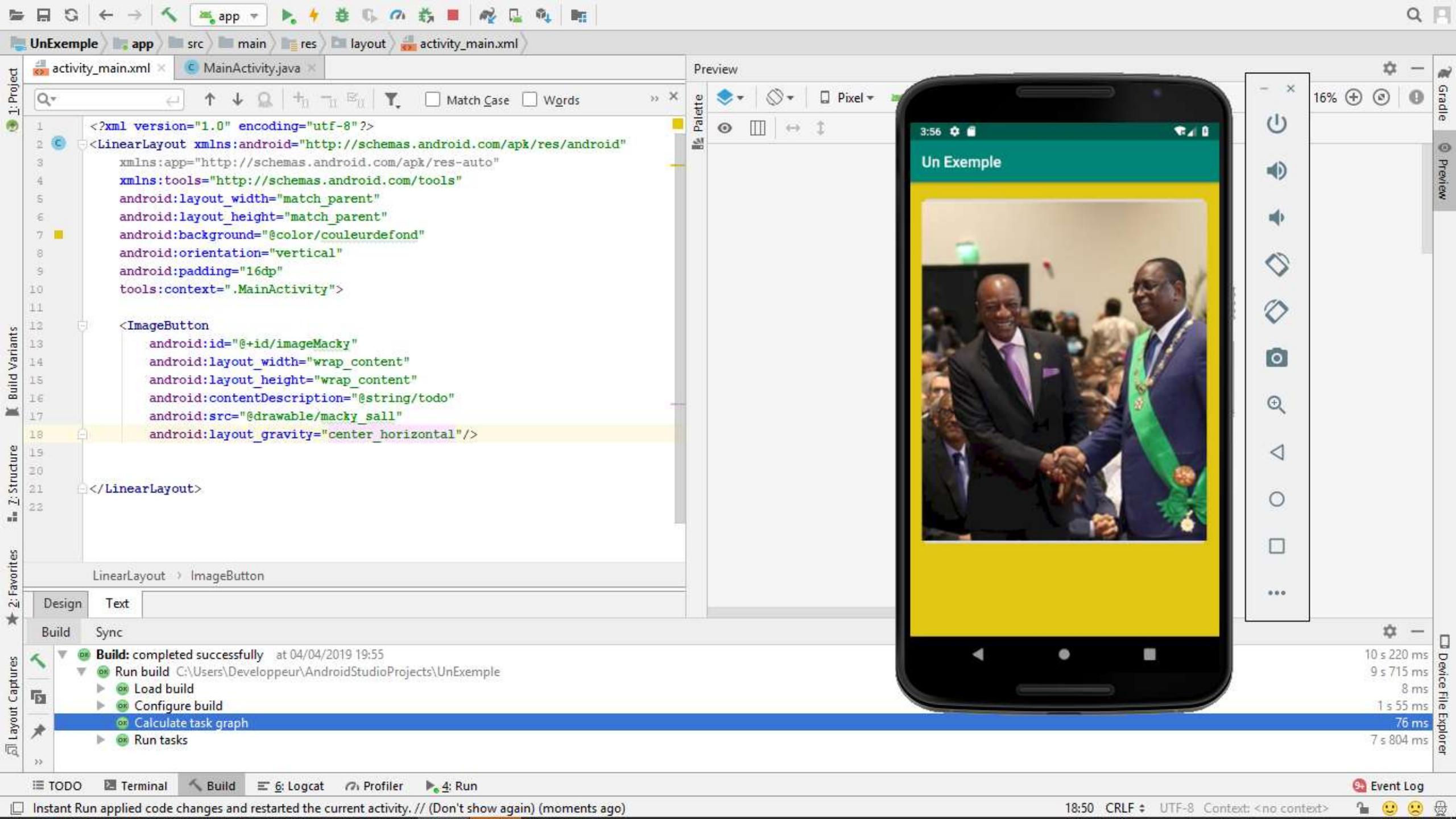
Build Sync

Build: completed successfully at 04/04/2019 19:55

- Run build C:\Users\Developpeur\AndroidStudioProjects\UnExemple
- Load build
- Configure build
- Calculate task graph
- Run tasks

Device File Explorer

Image of two men shaking hands on a smartphone screen.



# ImageView : Afficher des images

- Permet d'afficher des images
- Propriétés :
  - Contenu
    - android:src Pour définir une couleur ou une image.
    - android:tint Pour définir une couleur qui teinte l'image
  - Position et dimensions de l'image
    - android:adjustViewBounds La taille de l'ImageView sera ou pas modifiée
    - android:baselineAlignBottom Cadrage ou pas de l'image en bas de la zone
    - android:cropToPadding L'image sera ou pas coupée si elle est plus grande que la taille disponible
    - android:scaleType Pour définir le mode de redimensionnement de l'image avec ou sans déformation. (voir exemples transparent suivant)
  - Taille
    - android:maxHeight Pour définir la hauteur maximale
    - android:maxWidth Pour définir la largeur maximale

# ImageView : Afficher des images

```
<ImageView android:id="@+id/image"
    android:src="@drawable/keithwembley"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:maxHeight="200px"
    android:adjustViewBounds="true"
    android:scaleType="centerCrop"/>
```



```
<ImageView android:id="@+id/image"
    android:src="@drawable/keithwembley"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:maxHeight="200px"
    android:adjustViewBounds="true"
    android:scaleType="fitXY"/>
```



# Les éléments à deux états

- Ils ont les mêmes paramètres que TextView auxquels vient s'ajouter la définition de l'état initial :`android:checked="b"` où b vaut true ou false  
Pour définir l'état initial

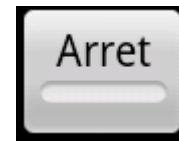
- CheckBox



- RadioButton



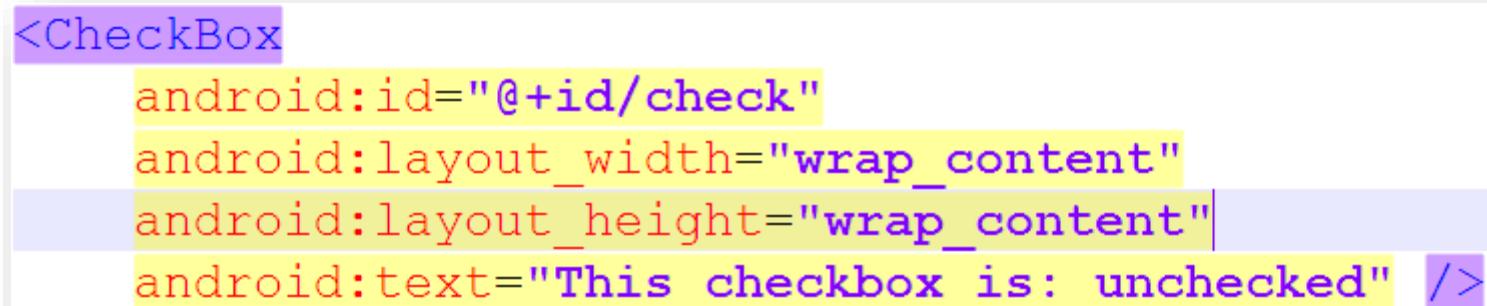
- ToggleButton



- `android:disabledAlpha` pour définir la transparence appliquée lorsque le bouton est inactif
- `android:textOff` Pour définir le texte quand le bouton n'est pas allumé
- `android:textOn` Pour définir le texte quand le bouton est allumé

# Les CheckBox

- Hérite de **CompoundButton**
- Possible d'alterer la valeur depuis le code:
  - **isChecked()**
  - **setChecked()**
  - **toggle()**
- Possible d'enregistrer un event listener **OnCheckedChangeListener**



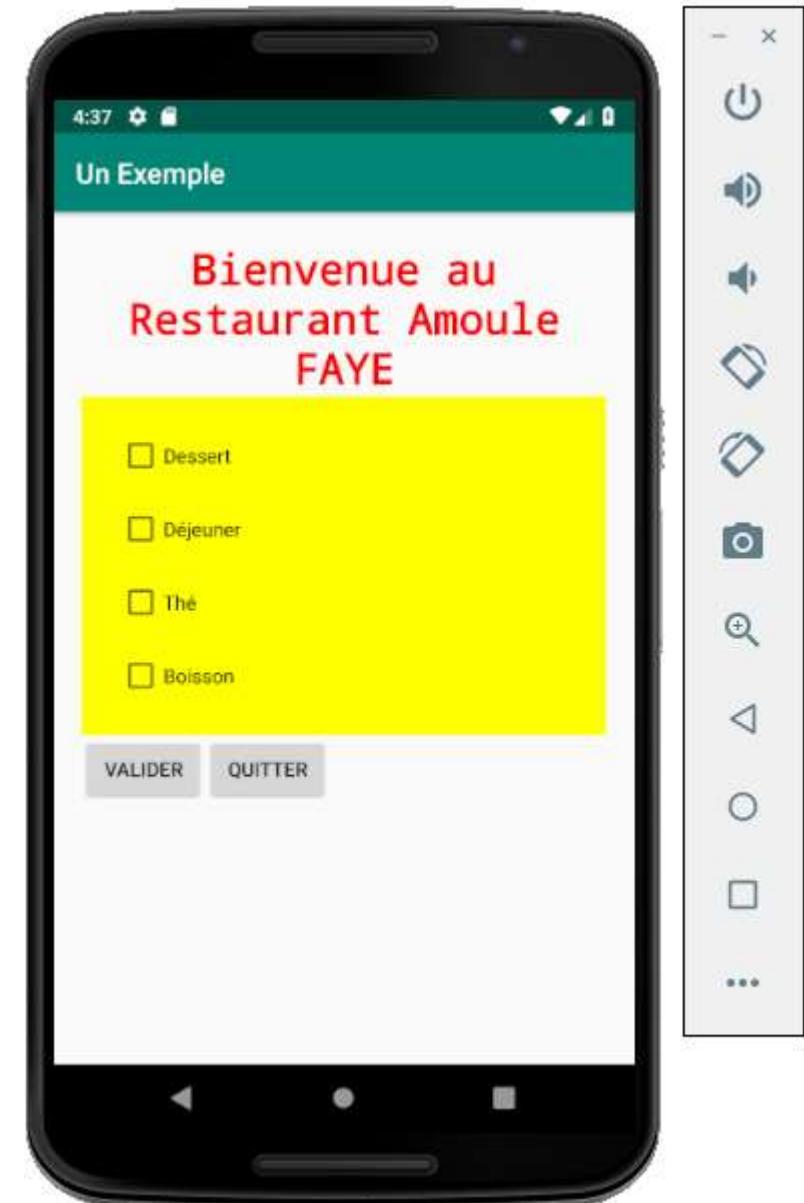
```
<CheckBox  
    android:id="@+id/check"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="This checkbox is: unchecked" />
```

# Exemple CheckBox

```
public class CheckBoxDemo extends Activity implements CompoundButton.OnCheckedChangeListener {  
    CheckBox cb;  
  
    @Override  
    public void onCreate(Bundle icicle) {  
        super.onCreate(icicle);  
        setContentView(R.layout.main);  
  
        cb=(CheckBox)findViewById(R.id.check);  
        cb.setOnCheckedChangeListener(this);  
    }  
  
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
        if (isChecked) {  
            cb.setText("Ce checkbox est: coché");  
        }  
        else {  
            cb.setText("Ce checkbox n'est pas coché");  
        }  
    }  
}
```

# CheckBox: exemple

- Un exemple d'utilisation d'un contrôle checkbox à l'intérieur de votre activité est le suivant :



# Basic vues: CheckBox

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     android:padding="20dp"
9     tools:context=".MainActivity">
10
11     <TextView
12         android:id="@+id/textView0"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:textColor="#FF0000"
16         android:gravity="center"
17         android:text="@string/message"
18         android:textSize="30sp"
19         android:textStyle="bold"
20         android:typeface="monospace"/>
21
22     <LinearLayout...>
23         <LinearLayout...>
24             ...
25         </LinearLayout>
26     </LinearLayout>
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68 </LinearLayout>
```

```
21 <LinearLayout
22     android:layout_width="match_parent"
23     android:layout_height="wrap_content"
24     android:background="#FFFF00"
25     android:orientation="vertical"
26     android:padding="16dp">
27     <CheckBox
28         android:id="@+id/dessert"
29         android:layout_width="wrap_content"
30         android:layout_height="wrap_content"
31         android:layout_margin="10dp"
32         android:text="Dessert"/>
33     <CheckBox
34         android:id="@+id/dejeuner"
35         android:layout_width="wrap_content"
36         android:layout_height="wrap_content"
37         android:layout_margin="10dp"
38         android:text="Déjeuner"/>
39     <CheckBox
40         android:id="@+id/the"
41         android:layout_width="wrap_content"
42         android:layout_height="wrap_content"
43         android:layout_margin="10dp"
44         android:text="Thé"/>
45     <CheckBox
46         android:id="@+id/boisson"
47         android:layout_width="wrap_content"
48         android:layout_height="wrap_content"
49         android:layout_margin="10dp"
50         android:text="Boisson"/>
51
52 </LinearLayout>
```

# Basic vues: CheckBox

The image shows two code editors side-by-side, likely from an IDE like Android Studio.

**Left Editor (Main Activity Layout):**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="20dp"
    tools:context=".MainActivity">

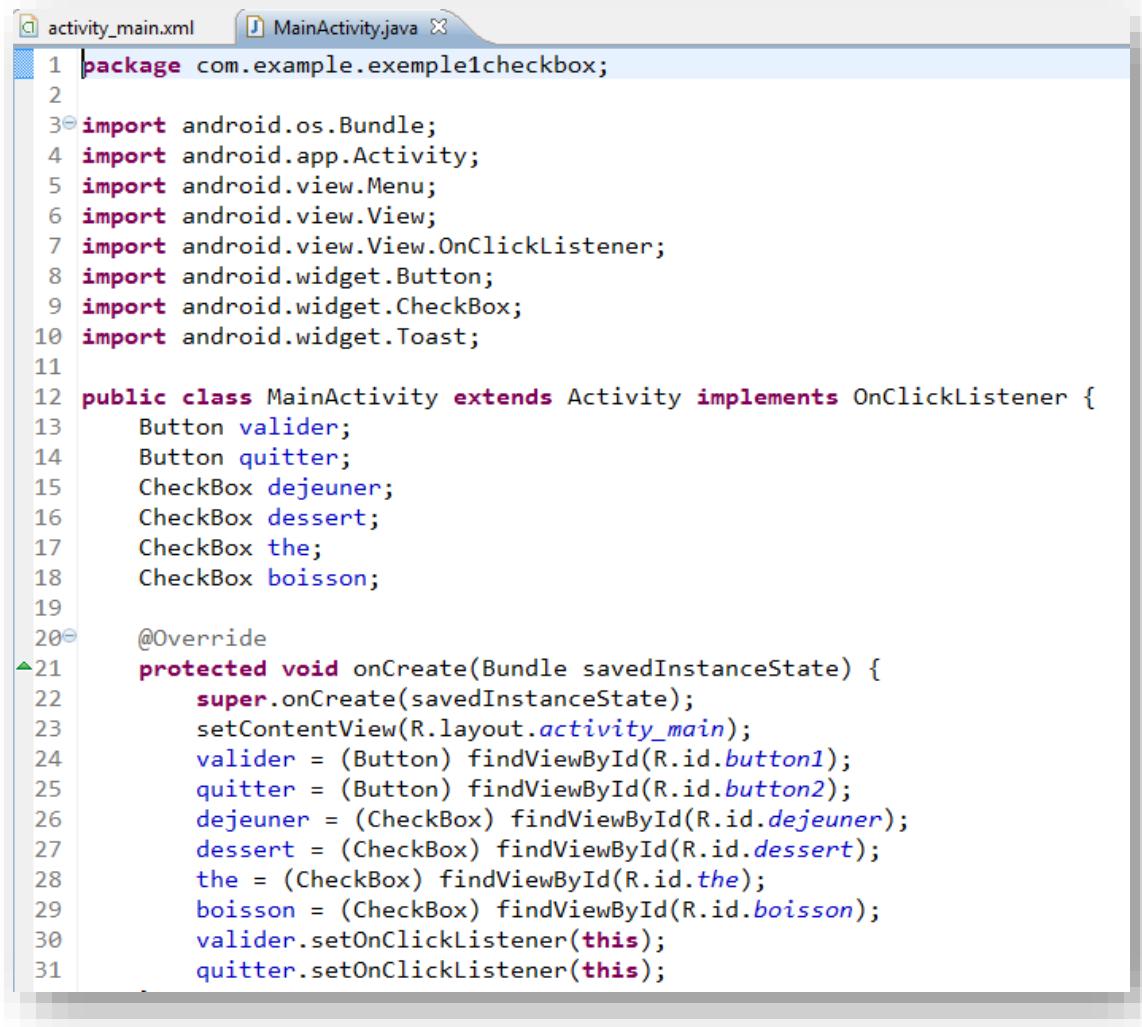
    <TextView
        android:id="@+id/textView0"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#FF0000"
        android:gravity="center"
        android:text="@string/message"
        android:textSize="30sp"
        android:textStyle="bold"
        android:typeface="monospace"/>

    <LinearLayout...>
    <LinearLayout...>
</LinearLayout>
```

**Right Editor (Fragment Layout):**

```
54 <LinearLayout
55     android:layout_width="match_parent"
56     android:layout_height="wrap_content">
57         <Button
58             android:id="@+id/boutonValider"
59             android:layout_width="wrap_content"
60             android:layout_height="wrap_content"
61             android:text="Valider" />
62         <Button
63             android:id="@+id/boutonQuitter"
64             android:layout_width="wrap_content"
65             android:layout_height="wrap_content"
66             android:text="Quitter" />
67     </LinearLayout>
```

# Basic vues: CheckBox



```
activity_main.xml MainActivity.java
1 package com.example.exemple1checkbox;
2
3 import android.os.Bundle;
4 import android.app.Activity;
5 import android.view.Menu;
6 import android.view.View;
7 import android.view.View.OnClickListener;
8 import android.widget.Button;
9 import android.widget.CheckBox;
10 import android.widget.Toast;
11
12 public class MainActivity extends Activity implements OnClickListener {
13     Button valider;
14     Button quitter;
15     CheckBox dejeuner;
16     CheckBox dessert;
17     CheckBox the;
18     CheckBox boisson;
19
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_main);
24         valider = (Button) findViewById(R.id.button1);
25         quitter = (Button) findViewById(R.id.button2);
26         dejeuner = (CheckBox) findViewById(R.id.dejeuner);
27         dessert = (CheckBox) findViewById(R.id.dessert);
28         the = (CheckBox) findViewById(R.id.the);
29         boisson = (CheckBox) findViewById(R.id.boisson);
30         valider.setOnClickListener(this);
31         quitter.setOnClickListener(this);
32     }
33 }
```

# Basic views: CheckBox

```
32     }
33
34     @Override
35     public boolean onCreateOptionsMenu(Menu menu) {
36         // Inflate the menu; this adds items to the action bar if it is present.
37         getMenuInflater().inflate(R.menu.activity_main, menu);
38         return true;
39     }
40
41     @Override
42     public void onClick(View v) {
43         if (v.getId() == valider.getId()) {
44             String[] commandes = new String[4];
45             int i = 0;
46             if (dejeuner.isChecked())
47                 commandes[i++] = "Déjeuner";
48             if (dessert.isChecked())
49                 commandes[i++] = "Dessert";
50             if (the.isChecked())
51                 commandes[i++] = "Thé";
52             if (boisson.isChecked())
53                 commandes[i++] = "Boisson";
54             if (i == 0) {
55                 Toast.makeText(getApplicationContext(),
56                     "Vous n'avez rien choisi !!!", Toast.LENGTH_LONG)
57                     .show();
58             } else {
59                 String str = "Vous avez commandé:\n";
60                 for (int j = 0; j < commandes.length; j++) {
61                     if (commandes[j] != null)
62                         str += "\t - " + commandes[j] + "\n";
63             }
64         }
65     }
66 }
```

# Basic vues: CheckBox

```
--  
63         }  
64         Toast.makeText(getApplicationContext(), str, Toast.LENGTH_LONG)  
65             .show();  
66     }  
67 } else {  
68     System.exit(0);  
69 }  
70 }  
71  
72 }
```

# RadioButton

- **RadioButton:** Mêmes paramètres XML que TextView plus android:checked="b" où b vaut true ou false coche ou décoche le bouton au départ
- Exemple de fichier XML :
  - <RadioButton android:id="@+id/gauche"
  - android:layout\_width="wrap\_content"
  - android:layout\_height="wrap\_content"
  - android:text="Left"
  - android:checked="false"
  - android:layout\_gravity="left"
  - />
- Méthodes de la classe RadioButton: android.widget.RadioButton
- Construction: RadioButton(Context) le paramètre est généralement l'activité elle-même
- Etat
  - isChecked() renvoie true si la case est cochée
  - setChecked(boolean) coche (true) ou décoche (false) la case
  - toggle() change l'état de la case

# RadioButtons

- Une case d'option est un bouton de deux États qui peut être soit cochée ou décochée.
- Si la case d'option est désactivée, l'utilisateur peut appuyer sur ou cliquez dessus pour le vérifier.
- Cases d'option sont normalement utilisés ensemble dans un RadioGroup.
- Lorsque plusieurs cases d'option sont à l'intérieur d'un groupe de boutons radio, la sélection d'une case d'option désactive toutes les autres.
- De même, vous pouvez appeler isChecked() sur un RadioButton pour voir si elle est activée, toggle() pour le sélectionner et ainsi de suite, comme vous pouvez le faire avec une case à cocher.

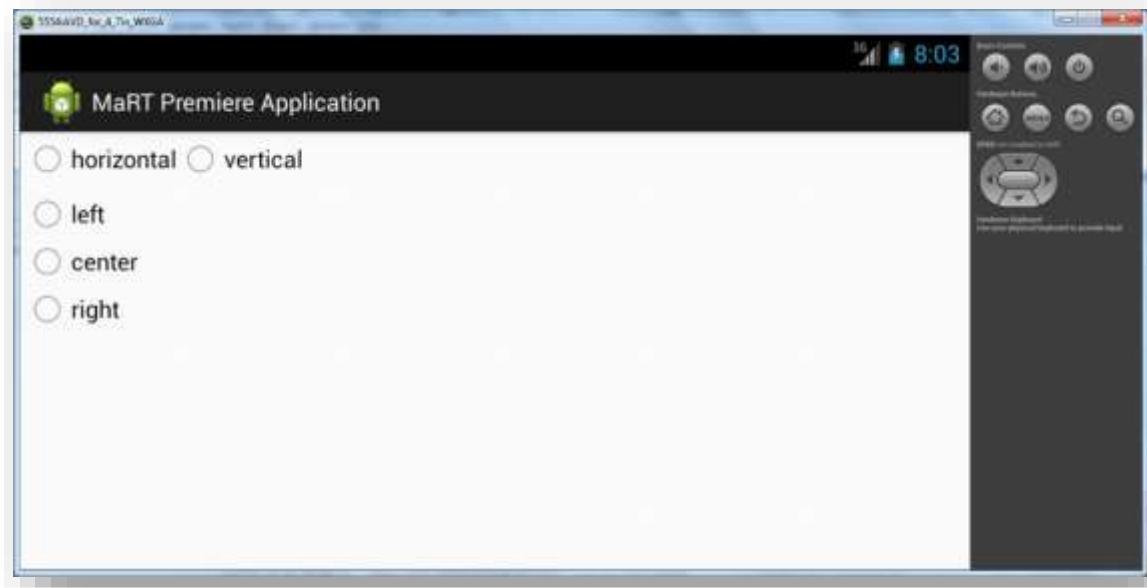
# RadioGroup

- Pour grouper des boutons radio (ajoute un ascenseur si nécessaire)
- Un seul bouton peut être coché à la fois (attention à l'état initial qui n'est pris en compte par le RadioGroup que s'il est fait par la méthode **check** du RadioGroup)
- Exemple de fichier XML :
  - <RadioGroup
  - android:id="@+id/groupe\_de\_boutons"
  - android:layout\_width="fill\_parent"
  - android:layout\_height="wrap\_content"
  - android:orientation="horizontal">
  - <RadioButton
  - ...
  - />
  - <RadioButton
  - ...
  - />
  - </RadioGroup>



# Exemple RadioGroup

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <RadioGroup android:id="@+id/orientation"
        android:orientation="horizontal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="5px">
        <RadioButton
            android:id="@+id/horizontal"
            android:text="horizontal" />
        <RadioButton
            android:id="@+id/vertical"
            android:text="vertical" />
    </RadioGroup>
    <RadioGroup android:id="@+id/gravity"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:padding="5px">
        <RadioButton
            android:id="@+id/left"
            android:text="left" />
        <RadioButton
            android:id="@+id/center"
            android:text="center" />
        <RadioButton
            android:id="@+id/right"
            android:text="right" />
    </RadioGroup>
</LinearLayout>
```

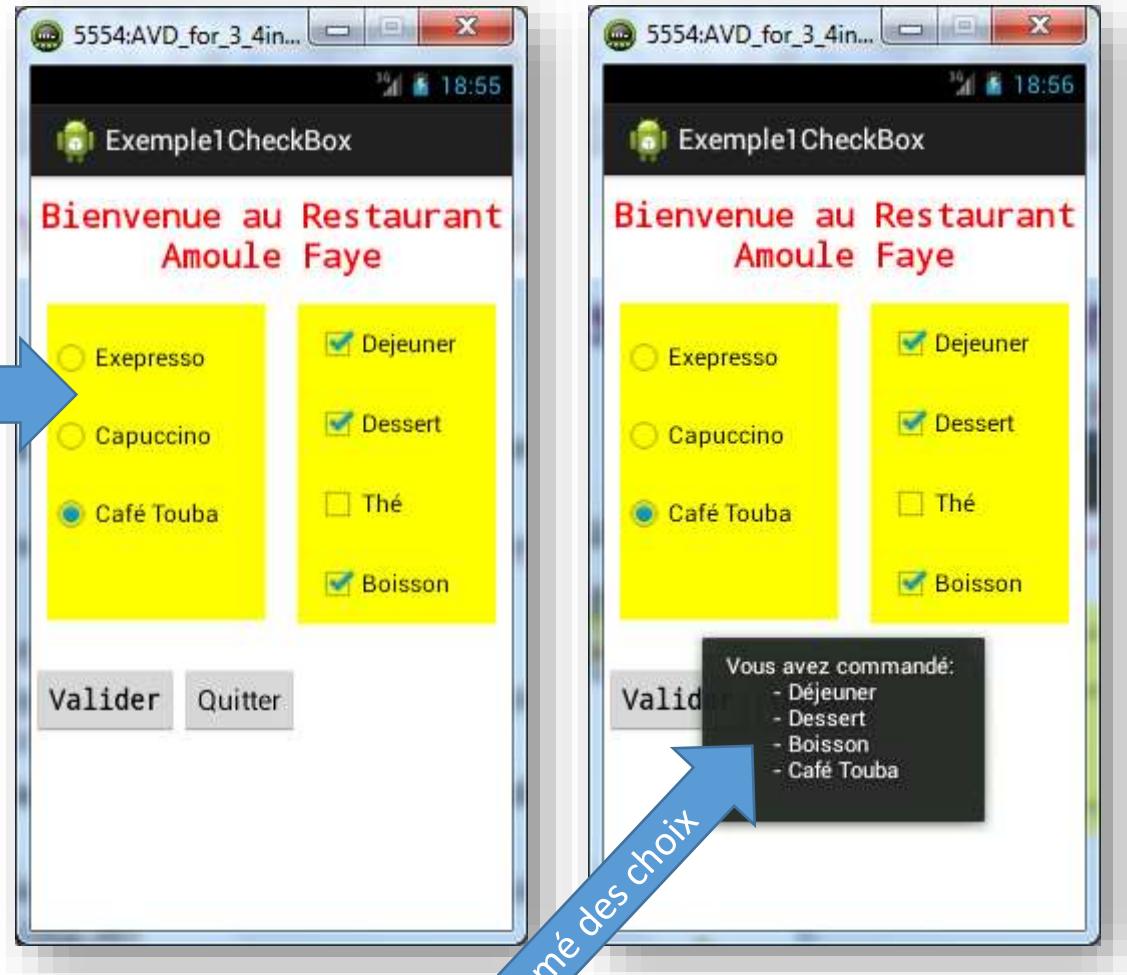


# Basic vues: RadioButtons

## Exemple

Cette interface utilisateur utilise des cases d'option et cases à cocher pour définir les choix

RadioGroup



# Basic vues: RadioButtons

- Nous étendons l'exemple précédent en ajoutant au RadioGroup trois RadioButton.
- Le nouveau code XML et Java seulement est produit :



```
activity_main.xml X
58      <LinearLayout
59          android:id="@+id/monPanneau"
60          android:layout_width="132dp"
61          android:layout_height="wrap_content"
62          android:layout_margin="10dp"
63          android:background="#FFFF00"
64          android:orientation="vertical" >
65
66          <CheckBox
67              android:id="@+id/dejeuner"
68              android:layout_width="wrap_content"
69              android:layout_height="wrap_content"
70              android:layout_margin="10dp"
71              android:text="@string/txtDejeuner" />
72
73          <CheckBox
74              android:id="@+id/dessert"
75              android:layout_width="wrap_content"
76              android:layout_height="wrap_content"
77              android:layout_margin="10dp"
78              android:text="@string/txtDessert" />
79
80          <CheckBox
81              android:id="@+id/the"
82              android:layout_width="wrap_content"
83              android:layout_height="wrap_content"
84              android:layout_margin="10dp"
85              android:text="@string/txtThe" />
86
87          <CheckBox
88              android:id="@+id/boisson"
89              android:layout_width="wrap_content"
90              android:layout_height="wrap_content"
91              android:layout_margin="10dp"
92              android:text="@string/txtBoisson" />
93
94      </LinearLayout>
95
96  </LinearLayout>
```

# Basic vues: RadioButtons

- Android Activity (1 of 3)

```
14 public class MainActivity extends Activity implements OnClickListener {  
15     Button valider;  
16     Button quitter;  
17     CheckBox dejeuner;  
18     CheckBox dessert;  
19     CheckBox the;  
20     CheckBox boisson;  
21     RadioGroup cafeRadioGroup;  
22     RadioButton expresso;  
23     RadioButton capuccino;  
24     RadioButton cafeTouba;
```

# Basic vues: RadioButtons

- Android Activity (2 of 3)

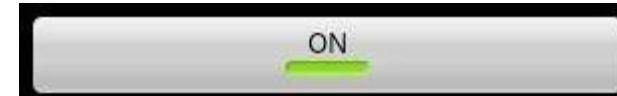
```
26 ⊖    @Override
27 ▲protected void onCreate(Bundle savedInstanceState) {
28     super.onCreate(savedInstanceState);
29     setContentView(R.layout.activity_main);
30     valider = (Button) findViewById(R.id.button1);
31     quitter = (Button) findViewById(R.id.button2);
32     dejeuner = (CheckBox) findViewById(R.id.dejeuner);
33     dessert = (CheckBox) findViewById(R.id.dessert);
34     the = (CheckBox) findViewById(R.id.the);
35     boisson = (CheckBox) findViewById(R.id.boisson);
36
37     cafeRadioGroup = (RadioGroup) findViewById(R.id.radioGroup1);
38     expresso = (RadioButton) findViewById(R.id.radio0);
39     capuccino = (RadioButton) findViewById(R.id.radio1);
40     cafeTouba = (RadioButton) findViewById(R.id.radio2);
41
42     valider.setOnClickListener(this);
43     quitter.setOnClickListener(this);
44 }
```

# Basic vues: RadioButtons

```
MainActivity.java X
53@Override
54    public void onClick(View v) {
55        if (v.getId() == valider.getId()) {
56            String[] commandes = new String[5];
57            int i = 0;
58            if (dejeuner.isChecked())
59                commandes[i++] = "Déjeuner";
60            if (dessert.isChecked())
61                commandes[i++] = "Dessert";
62            if (the.isChecked())
63                commandes[i++] = "Thé";
64            if (boisson.isChecked())
65                commandes[i++] = "Boisson";
66            if (expresso.getId() == cafeRadioGroup.getCheckedRadioButtonId()) {
67                commandes[i++] = "Expresso";
68            }
69            if (capuccino.getId() == cafeRadioGroup.getCheckedRadioButtonId()) {
70                commandes[i++] = "Capucino";
71            }
72            if (cafeTouba.getId() == cafeRadioGroup.getCheckedRadioButtonId()) {
73                commandes[i++] = "Café Touba";
74            }
75
76            if (i == 0) {
77                Toast.makeText(getApplicationContext(),
78                    "Vous n'avez rien choisi !!!", Toast.LENGTH_LONG)
79                    .show();
80            } else {
81                String str = "Vous avez commandé:\n";
82                for (int j = 0; j < commandes.length; j++) {
83                    if (commandes[j] != null)
84                        str += "\t - " + commandes[j] + "\n";
85                }
86                Toast.makeText(getApplicationContext(), str, Toast.LENGTH_LONG)
87                    .show();
88            }
89        } else {
90            System.exit(0);
91        }
92    }
93}
```

# ToggleButton

- Mêmes paramètres XML que TextView plus :
  - android:disabledAlpha="x" (où x est une valeur réelle entre 0 et 1) définit la transparence appliquée lorsque le bouton est inactif
  - android:textOff="txt" définit le texte quand le bouton n'est pas allumé
  - android:textOn="txt" définit le texte quand le bouton est allumé
  - android:checked="b" (où b vaut true ou false) place le bouton en position allumé ou éteint
- **Méthodes de la classe ToggleButton:** android.widget.ToggleButton
- **Construction:** ToggleButton(Context) le paramètre est généralement l'activité elle-même
- **Etat**
  - isChecked() indique si le bouton est allumé ou éteint (true/false)
  - setChecked(boolean) positionne le bouton (true = allumé, false = éteint)
  - toggle() change l'état du bouton
- **Aspect**
  - getTextOff() renvoie le texte associé au bouton quand il n'est pas allumé
  - getTextOn() renvoie le texte associé au bouton quand il est allumé
  - setTextOff(CharSequence) définit le texte associé au bouton quand il n'est pas allumé
  - setTextOn(CharSequence) définit le texte associé au bouton quand il est allumé



# Autocomplétion : AutoCompleteTextView

- C'est une spécialisation de EditText pour apporter l'auto complétion
- Propriétés supplémentaires:
  - android:completionHint="texte" texte affiché en titre du menu déroulant
  - android:completionThreshold Pour définir le nombre de caractères à taper avant que la complétion n'entre en action.
  - android:dropDownHeight="unité" on peut aussi utiliser les constantes fill\_parent et wrap\_content, définit la hauteur du menu déroulant
  - android:dropDownWidth="unité" on peut aussi utiliser les constantes fill\_parent et wrap\_content, définit la hauteur du menu déroulant
  - android:dropDownHorizontalOffset Pour définir le décalage horizontal du menu déroulant
  - android:dropDownVerticalOffset Pour définir le décalage vertical du menu déroulant

# Autocomplétion : AutoCompleteTextView

- **Construction**
  - AutoCompleteTextView(Context) le paramètre est généralement l'activité elle-même
- **Aspect et contenu**
  - showDropDown() montre la liste déroulante
  - dismissDropDown() cache la liste déroulante
  - getListSelection() renvoie le rang de la proposition choisie
  - setListSelection(int) choisit une proposition par son rang
  - setAdapter(ArrayAdapter) Permet de remplir la liste de propositions. La classe ArrayAdapter est décrite dans ListView.
  - setCompletionHint(CharSequence) définit le texte affiché en titre du menu déroulant

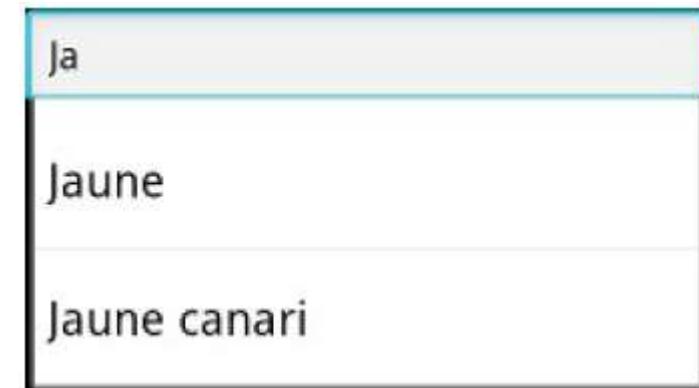
# Autocomplétion: exemple

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <AutoCompleteTextView
        android:id="@+id/complete"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

# Autocomplétion: exemple

- Déclarer l'activité AutoCompletionActivity suivante :

```
import android.app.Activity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
public class AutoCompletionActivity extends Activity {
    private AutoCompleteTextView complete = null;
    // Notre liste de mots que connaîtra l'AutoCompleteTextView
    private static final String[] COULEUR = new String[] {
        "Bleu", "Vert", "Jaune", "Jaune canari", "Rouge", "Orange"
    };
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        // On récupère l'AutoCompleteTextView déclaré dans notre layout
        complete = (AutoCompleteTextView) findViewById(R.id.complete);
        complete.setThreshold(2);
        // On associe un adaptateur à notre liste de couleurs...
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
            android.R.layout.simple_dropdown_item_1line, COULEUR);
        // ... puis on indique que notre AutoCompleteTextView utilise cet adaptateur
        complete.setAdapter(adapter);
    }
}
```



# GESTION DES ÉVÉNEMENTS

# Événements

Quelque chose qui arrive

- Dans l'interface utilisateur: Cliquez, tapez, faites glisser
- Appareil: DetectedActivity, comme marcher, conduire, basculer
- Les événements sont "remarqués" par le système Android

# Gestionnaires d'événements

Méthodes qui font quelque chose en réponse à un clic

Une méthode, appelée **gestionnaire d'événements**, est déclenchée par un événement spécifique et agit en réponse à cet événement.

# La gestion des événements

- la programmation des applications devient intéressant quand on peut capturer les actions de l'utilisateur pour interagir avec lui.
- Sous Android, les Listener permettent de spécifier le comportement des éléments en fonction des évènements que l'utilisateur génère. Les principaux Listeners sont: **onClickListener**, **onLongClickListener** , **onTouchListener**, **onCreateContextMenuListener**, **onFocusChangeListener**, **onKeyListener...**

# La gestion des événements

- **onClickListener** - Utilisé pour détecter des événements click qui correspond à un appui suivi d'un relâchement d'une zone de l'écran qui contient une vue. Ce Listener permet créer un CallBack sur l'événement onClick().
- **onLongClickListener** - Utilisé pour détecter lorsque l'utilisateur maintient le contact sur une vue pendant une période prolongée. Correspond à la méthode de rappel onLongClick () qui est passé à la vue qui a reçu l'événement comme un argument.

# La gestion des événements

- **onTouchListener** - utilisés pour détecter toute forme de contact avec l'écran tactile, y compris des touches individuelles ou multiples et des mouvements gestuels. Correspondant à la fonction OnTouch().
- **onCreateContextMenuListener** - Attend la création d'un menu contextuel comme le résultat d'un long clic. Correspond à la méthode de rappel onCreateContextMenu () .

# La gestion des événements

- **onFocusChangeListener** - Déetecte lorsque le focus se déplace vers une autre vue à la suite d'une interaction avec un track-ball ou la touche de navigation. Correspond à la onFocusChange () .
- **onKeyListener** - Utilisé pour détecter quand une touche sur le clavier est enfoncé. Correspond à la méthode OnKey () .

# Attacher en XML et implémenter en Java

**Attachez le gestionnaire à la vue en format XML:**

    android:onClick="showToast"

**Implémenter le Gestionnaire d'événement dans l'activité Java:**

```
public void showToast(View view) {  
    String msg = "Hello Toast!";  
    Toast toast =Toast.makeText(this,  
                                msg, duration);  
    toast.show();  
}
```

# Alternative: Définir le gestionnaire de clics en Java

```
final Button button = (Button) findViewById(R.id.button_id);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        String msg = "Hello Toast!";
        Toast toast = Toast.makeText(getApplicationContext(),
                msg, duration);
        toast.show();
    }
});
```

# La gestion des événements

- Un exemple de gestion de l'événement est illustré par ce bouton.

```
Button button = (Button)findViewById(R.id.myButton);

button.setOnClickListener(
    new Button.OnClickListener() {
        public void onClick(View v) {
            TextView myTextView =
                (TextView)findViewById(R.id.myTextView);
            myTextView.setText("A click on the button");
        }
    });
;

button.setOnLongClickListener(
    new Button.OnLongClickListener() {
        public boolean onLongClick(View v) {
            TextView myTextView =
                (TextView)findViewById(R.id.myTextView);
            myTextView.setText("A long click on the button");
            return true;
        }
    });
;
```

# Evaluation

- Exercice 1: Browse documentation concerning TextView widget at <http://developer.android.com/reference/android/widget/TextView> and fill in the following table:

Property	Description
android:autoLink	
android:capitalize	
android:editable	
android:fontFamily	
android:gravity	
android:text	
android:textSize	
android:width	
android:height	

- Find in the same page, methods that allow retrieving and modifying value of the attributes in java program.

# Evaluation

- **Exercice 2:** Ecrivez une application “changez la couleur de fond” qui contient un EditText et un Button. Chaque clic sur le bouton changera de manière aléatoire la couleur de fond de EditText.

# Les groupes

- Regrouper des éléments participant à un choix:
  - **ListView** (plusieurs éléments organisés en liste verticale avec séparateurs). Souvent utilisé pour des listes de mots (type menu).
  - **GridView** (plusieurs éléments organisés en table). Souvent utilisé pour des tables de mots (type menu).
  - **RadioGroup** (groupe de boutons radio dont un seul peut être coché à la fois)- Voir partie précédente
  - **Gallery** (plusieurs éléments organisées horizontalement avec défilement). Souvent utilisé pour des images

# ListView

- Place les éléments en liste verticale et ajoute un ascenseur si nécessaire
  - **Séparateurs**
    - **android:divider** Pour définir la couleur des séparateurs ou pour utiliser une image comme séparateur.
    - **android:dividerHeight="unité"** Pour définir la hauteur des séparateurs (même s'ils contiennent une image)
  - **Type de choix**
    - **android:choiceMode="c"** (où c peut prendre les valeurs : **none**, **singlechoice**, **multipleChoice**) pour indiquer le mode de choix dans la liste (aucun, un seul, plusieurs).

# ListView (Contenu)

- ***En XML (texte seulement)***
  - `android:entries="@array/maliste"` définit le contenu de la liste à partir du contenu d'un fichier xml placé dans res/values/ et qui a la forme :
- ***Dans le code (éléments quelconques)***
- ***On utilise un gestionnaire de contenu (Adapter)***
  - `setAdapter(Adapter)` pour associer à la ListView
  - ***Soit de classe prédefinie (`ArrayAdapter` , `SimpleAdapter`, `CursorAdapter`)***
    - `ArrayAdapter(Context, type)` le second paramètre est une type prédefini :
      - `android.R.layout.simple_list_item_1` pour une liste à choix unique ou
      - `android.R.layout.simple_list_item_multiple_choice` pour une liste à choix multiple(une case à cocher apparaît à coté de chaque élément de la liste)
    - `ArrayAdapter.add(Object)` pour remplir la liste
  - ***Soit de classe personnalisée (héritage de `BaseAdapter`)***

# Une précision à propos des adaptateurs

- Les adaptateurs sont des classes qui lient des données aux vues de l'UI
  - Les vues concernées étendent android.widget.AdapterView
- Classes d'adaptateurs
  - Héritent de android.widget.BaseAdapter
  - SimpleAdapter, ArrayAdapter<?> : sert à récupérer des données stockées dans une collection
- Exploite par défaut la valeur de la méthode `toString()` des objets de la liste
  - CursorAdapter : sert à récupérer des données stockées dans une base de données relationnelle (SQLite)
  - Vous pouvez étendre ces classes de base pour gérer finement vos items (conseillé)

# Exemple de ListView

1. Créer une nouvelle activité vide ou un nouveau projet
2. Placez un conteneur ListView sur son fichier de disposition
3. Bouton Ajouter pour validation

ExempleaveclesListes app src main res values

Android 1: Project

app manifests java com.example.exempleavecleslistes MainActivity com.example.exempleavecleslistes (an com.example.exempleavecleslistes (te generatedJava res drawable layout mipmap values Gradle Scripts

Build Variants Favorites Layout Captures

Palette Common Text Buttons Widgets Layouts Containers Google Legacy

activity\_main.xml arrays.xml MainActivity.java

Pixel 28 AppTheme 19% Attributes

id listeEtudiants layout\_width match\_parent layout\_height wrap\_content

Layout\_Margin [?, ?, ?, ?, ?]

Padding [?, ?, ?, ?, ?]

Theme choiceMode multipleChoice addStatesFrom alpha alwaysDrawnW animateLayout animationCach background cacheColorHint clickable clipChildren clipToPadding contentDescrip descendantFoc divider dividerHeight drawSelectorOr drawingCacheC duplicateParent entries fadeScrollbars fadingEdge

Component Tree

LinearLayout(vertical)

listeEtudiants

button2- "Valider votr..."

Item 1 Sub Item 1  
Item 2 Sub Item 2  
Item 3 Sub Item 3  
Item 4 Sub Item 4  
Item 5 Sub Item 5  
Item 6 Sub Item 6  
Item 7 Sub Item 7  
Item 8 Sub Item 8  
Item 9 Sub Item 9  
Item 10 Sub Item 10  
Item 11 Sub Item 11

Device File Explorer

Design

Text

TODO

Terminal

Build

Logcat

Profiler

Run

Event Log



ExempleaveclesListes app src main res layout activity\_main.xml

Structure  
1: Project  
LinearLayout  
ListView @+id/listeEtudiants  
Button @+id/button2

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/ap
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     tools:context=".MainActivity">
9
10    <ListView
11        android:id="@+id/listeEtudiants"
12        android:layout_width="match_parent"
13        android:layout_height="wrap_content"
14        android:choiceMode="multipleChoice" />
15
16    <Button
17        android:id="@+id/button2"
18        android:layout_width="match_parent"
19        android:layout_height="wrap_content"
20        android:onClick="validerChoix"
21        android:text="Valider votre choix" />
22
23 </LinearLayout>
```

LinearLayout ListView

Preview



Item 1	Sub Item 1
Item 2	Sub Item 2
Item 3	Sub Item 3
Item 4	Sub Item 4
Item 5	Sub Item 5
Item 6	Sub Item 6
Item 7	Sub Item 7
Item 8	Sub Item 8
Item 9	Sub Item 9
Item 10	Sub Item 10
Item 11	Sub Item 11
...	...

Gradle Preview

Device File Explorer



1: Project

Palette

Common Text Buttons Widgets Layouts Containers Google Legacy

2: Favorites

Component Tree

LinearLayout(vertical)  
listEtudiants  
button2- "Valider votre ch...

Resources

Drawable android config\_keySystem  
Color emailAddressType Home  
ID imProtocols AIM  
String organizationType Work  
Style phoneTypes Home  
postalAddressType Home

New Array Value Resource

Resource name: etudiants  
Resource value:  
Source set: main  
File name: arrays.xml  
Create the resource in directories:  
values  OK Cancel

Attributes

clipChildren  
clipToPadding  
contentDescr...  
descendantFoc...  
divider  
dividerHeight  
drawSelectorOr...  
drawingCacheC...  
duplicateParent...  
entries  
fadeScrollbars...  
fadingEdge  
fadingEdgeLen...  
fastScrollAlway...  
fastScrollEnable...  
filterTouchesW...  
fitsSystemWind...  
focusable  
focusableInTou...  
footerDividersE...  
foreground  
foregroundGrav...  
hapticFeedback...  
headerDividersE...  
isScrollContain...  
keepScreenOn...  
layerType

ExempleaveclesListes [C:\Users\Developpeur\AndroidStudioProjects\ExempleaveclesListes] - ...\\app\\src\\main\\res\\values\\arrays.xml

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

ExempleaveclesListes app src main res values arrays.xml

Android app manifests java com.example.exempleavecleslistes MainActivity generatedJava res drawable layout mipmap values arrays.xml colors.xml strings.xml styles.xml Gradle Scripts

activity\_main.xml arrays.xml MainActivity.java

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="etudiants">
        <item>Moussa FALL</item>
        <item>Mamadou FAYE</item>
        <item>Samba TALL</item>
        <item>Abdou KANE</item>
        <item>Yacine KA</item>
        <item>Modou POUYE</item>
        <item>Bamba FALL</item>
        <item>Racine KANE</item>
        <item>Awa GUEYE</item>
    </string-array>
</resources>
```

resources > string-array

Logcat TODO Terminal Build Profiler Run Event Log

Instant Run applied code changes and restarted the app. // (Don't ... (20 minutes ago) 7:6 CRLF UTF-8 Context: <no context>

activity\_main.xml arrays.xml MainActivity.java

Project

1: Project

2: Structure

3: Favorites

4: Build Variants

5: Layout Captures

6: Logcat

7: TODO

8: Terminal

9: Build

10: Profiler

11: Run

I: Structure

2: Favorites

Build Variants

Layout Captures

Logcat

TODO

Terminal

Build

Profiler

Run

MainActivity

&gt; onCreate()

11

public class MainActivity extends AppCompatActivity {

12     ListView liste;

13     String[] etudiants;

14     @Override

15     protected void onCreate(Bundle savedInstanceState) {

16         super.onCreate(savedInstanceState);

17         setContentView(R.layout.activity\_main);

18         liste = (ListView) findViewById(R.id.listeEtudiants);

19         etudiants = getResources().getStringArray(R.array.etudiants);

20         ArrayAdapter&lt;String&gt; adapteur = new ArrayAdapter&lt;String&gt;(context: this,

21             android.R.layout.simple\_list\_item\_multiple\_choice);

22         for (int i = 0; i &lt; etudiants.length; i++)

23             adapteur.add(etudiants[i]);

24         liste.setAdapter(adapteur);

25     }

26     public void validerChoix(View v) {

27         SparseBooleanArray selection = liste.getCheckedItemPositions();

28         String choix="Vous avez choisi:";

29         for (int i = 0; i &lt; selection.size(); i++) {

30             choix+="\n - "+etudiants[selection.keyAt(i)];

31         }

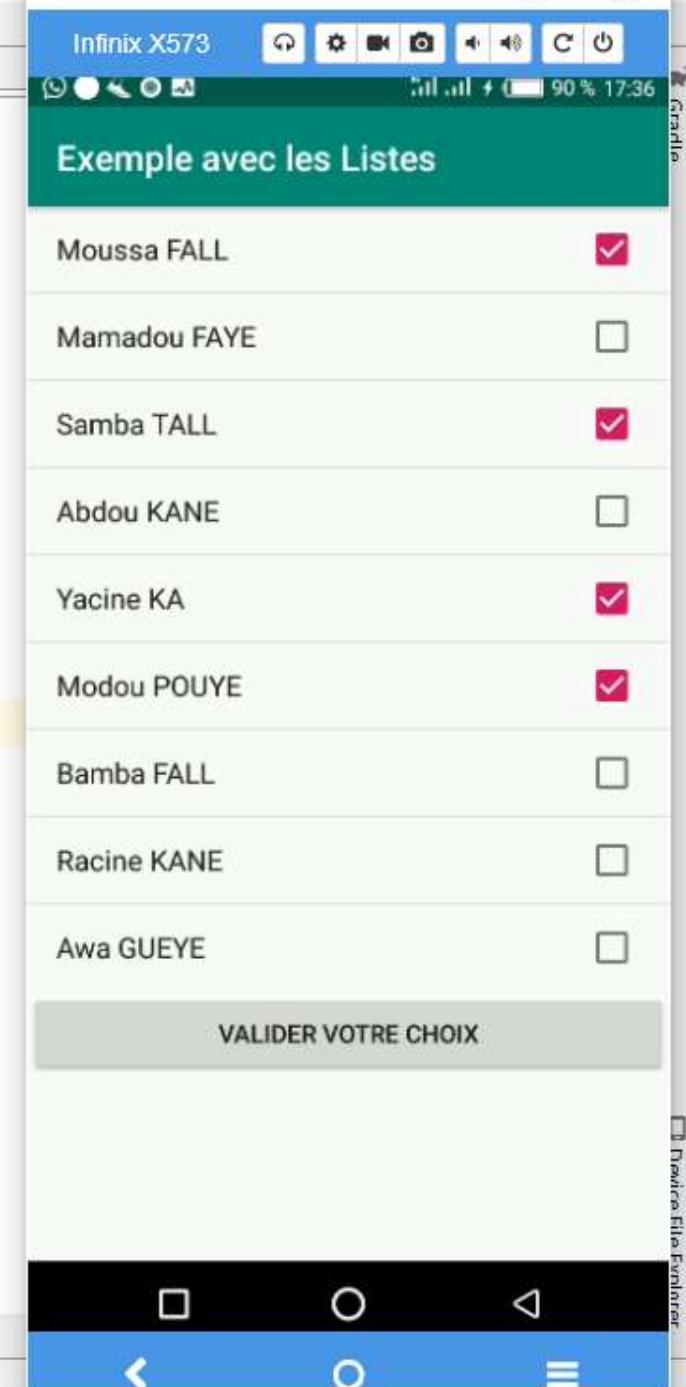
32         Toast.makeText(context: this, choix, Toast.LENGTH\_LONG).show();

33         liste.clearChoices();

34         ((ArrayAdapter) liste.getAdapter()).notifyDataSetChanged();

35     }

36 }



activity\_main.xml arrays.xml MainActivity.java

1: Project

2: Structure

3: Favorites

4: Build Variants

5: Layout Captures

6: Logcat

7: TODO

8: Terminal

9: Build

10: Profiler

11: Run

12: Layout Inspector

13: Device File Explorer

14: Find/Replace

15: References

16: Symbol

17: Layout Preview

18: Layout Inspector

19: References

20: Symbol

21: Layout Preview

22: References

23: Symbol

24: Layout Preview

25: References

26: Symbol

27: Layout Preview

28: References

29: Symbol

30: Layout Preview

31: References

32: Symbol

33: Layout Preview

34: References

35: Symbol

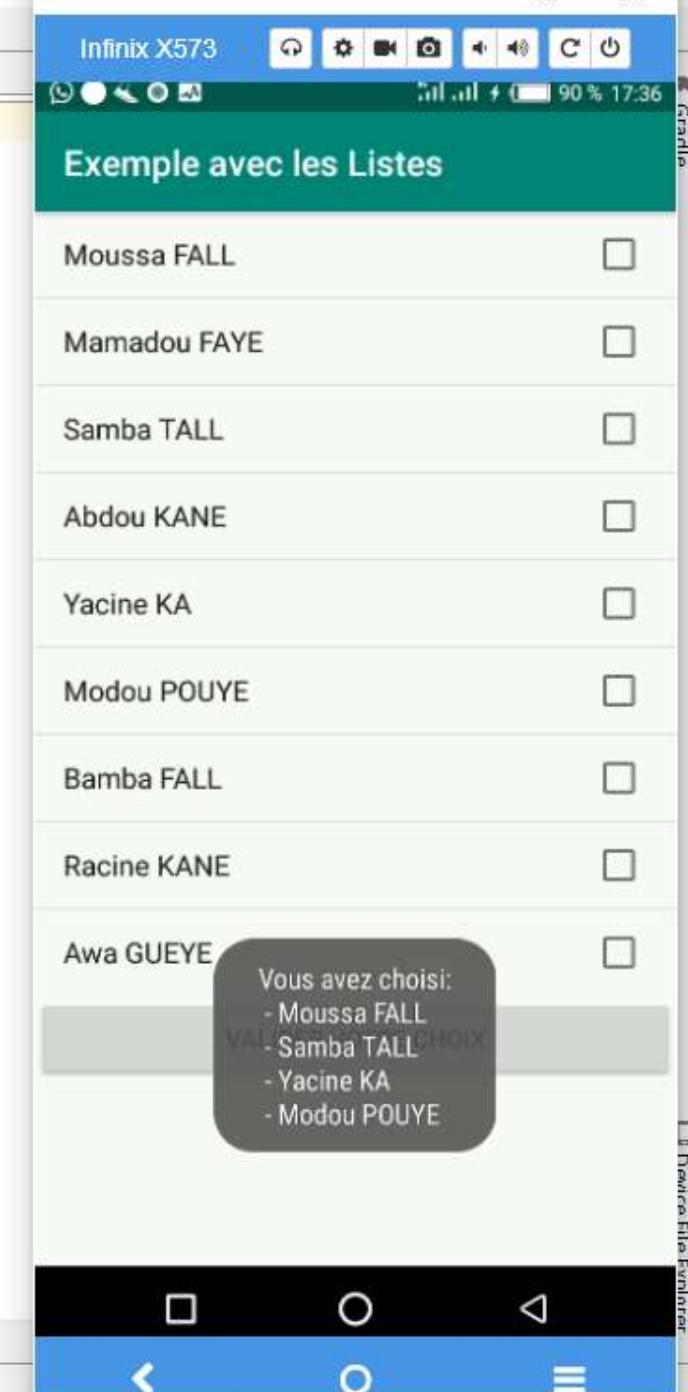
36: Layout Preview

MainActivity

```

11 public class MainActivity extends AppCompatActivity {
12     ListView liste;
13     String[] etudiants;
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18         liste = (ListView) findViewById(R.id.listeEtudiants);
19         etudiants = getResources().getStringArray(R.array.etudiants);
20         ArrayAdapter<String> adapteur = new ArrayAdapter<String>(context: this,
21             android.R.layout.simple_list_item_multiple_choice);
22         for (int i = 0; i < etudiants.length; i++)
23             adapteur.add(etudiants[i]);
24         liste.setAdapter(adapteur);
25     }
26     public void validerChoix(View v) {
27         SparseBooleanArray selection = liste.getCheckedItemPositions();
28         String choix="Vous avez choisi:";
29         for (int i = 0; i < selection.size(); i++) {
30             choix+="\n - "+etudiants[selection.keyAt(i)];
31         }
32         Toast.makeText(context: this, choix, Toast.LENGTH_LONG).show();
33         liste.clearChoices();
34         ((ArrayAdapter) liste.getAdapter()).notifyDataSetChanged();
35     }
36 }

```



# Liste de choix (Spinner)

- Affiche le choix actuel et affiche un RadioGroup quand on clique dessus pour le changer
- Propriétés :
  - **android:prompt** Pour définir le titre de la fenêtre qui s'ouvre lorsque l'on fait un choix
  - **android:entries="@array/malist e"** définit le contenu de la liste à partir du contenu du fichier string.xml placé dans res/values/ qui a la forme du fichier en face :

1: Project

activity\_main.xml arrays.xml MainActivity.java

Preview

Palette

Gradle Preview

Device File Explorer

1: Structure

2: Favorites

Build Variants

Layout Captures

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    android:layout\_margin="16dp"  
    android:orientation="vertical"  
    tools:context=".MainActivity">

<LinearLayout  
    android:layout\_width="match\_parent"  
    android:layout\_height="wrap\_content"  
    android:orientation="horizontal">

<TextView  
    android:id="@+id/textView"  
    android:layout\_width="wrap\_content"  
    android:layout\_height="wrap\_content"  
    android:text="Choix ?" />

<Spinner  
    android:id="@+id/listeEtudiants"  
    android:layout\_width="match\_parent"  
    android:layout\_height="wrap\_content"  
    android:addStatesFromChildren="false"  
    android:choiceMode="singleChoice"  
    android:entries="@array/etudiants" />

</LinearLayout>

<Button  
    android:id="@+id/button2"  
    android:layout\_width="match\_parent"  
    android:layout\_height="wrap\_content"  
    android:onClick="validerChoix"  
    android:text="Valider votre choix" />

</LinearLayout>

LinearLayout > LinearLayout > Spinner

Design Text

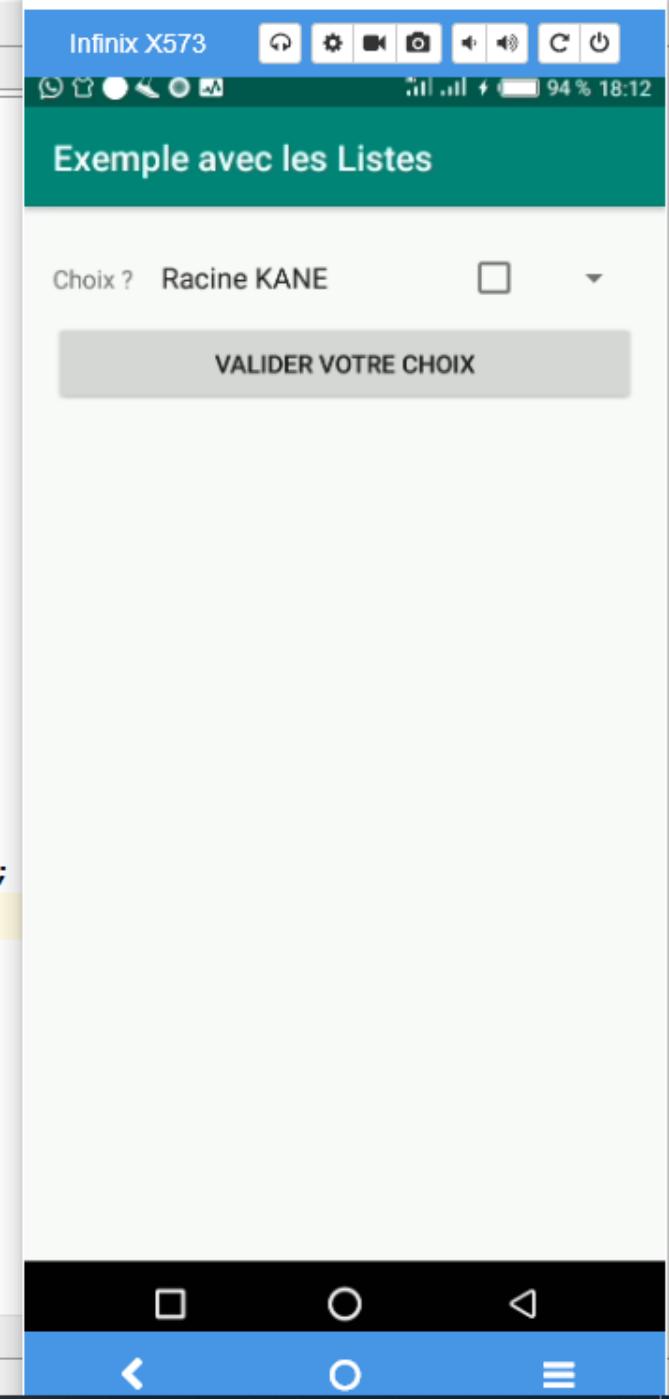
6 Logcat TODO Terminal Build Profiler 4 Run Event Log

The screenshot shows the Android Studio interface for a project named "ExempleaveclesListes". The main window displays the XML layout file "activity\_main.xml" in the "Text" tab. The XML code defines a vertical LinearLayout containing a horizontal LinearLayout with a TextView and a Spinner, and a Button at the bottom. The "Design" tab is visible at the bottom left. On the right side, there are two preview panes: the top one shows a light blue-themed activity with a spinner dropdown menu showing "Moussa FALL" and a button labeled "VALIDER VOTRE CHOIX"; the bottom one shows a dark blue-themed activity with the same button. The left sidebar includes sections for Project, Structure, Favorites, Build Variants, and Layout Captures.

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

activity\_main.xml arrays.xml MainActivity.java

```
12 public class MainActivity extends AppCompatActivity {
13     Spinner liste;
14     String[] etudiants;
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19         liste = (Spinner) findViewById(R.id.listeEtudiants);
20         etudiants = getResources().getStringArray(R.array.etudiants);
21         ArrayAdapter<String> adapteur = new ArrayAdapter<String>(context: this,
22             android.R.layout.simple_list_item_multiple_choice);
23         for (int i = 0; i < etudiants.length; i++)
24             adapteur.add(etudiants[i]);
25         liste.setAdapter(adapteur);
26     }
27     public void validerChoix(View v) {
28         String choix="Vous avez choisi "+etudiants[liste.getSelectedItemPosition()];
29         Toast.makeText(context: this, choix, Toast.LENGTH_LONG).show();
30     }
31 }
32
```

1: Project  
2: Favorites  
Build Variants  
Layout Captures

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

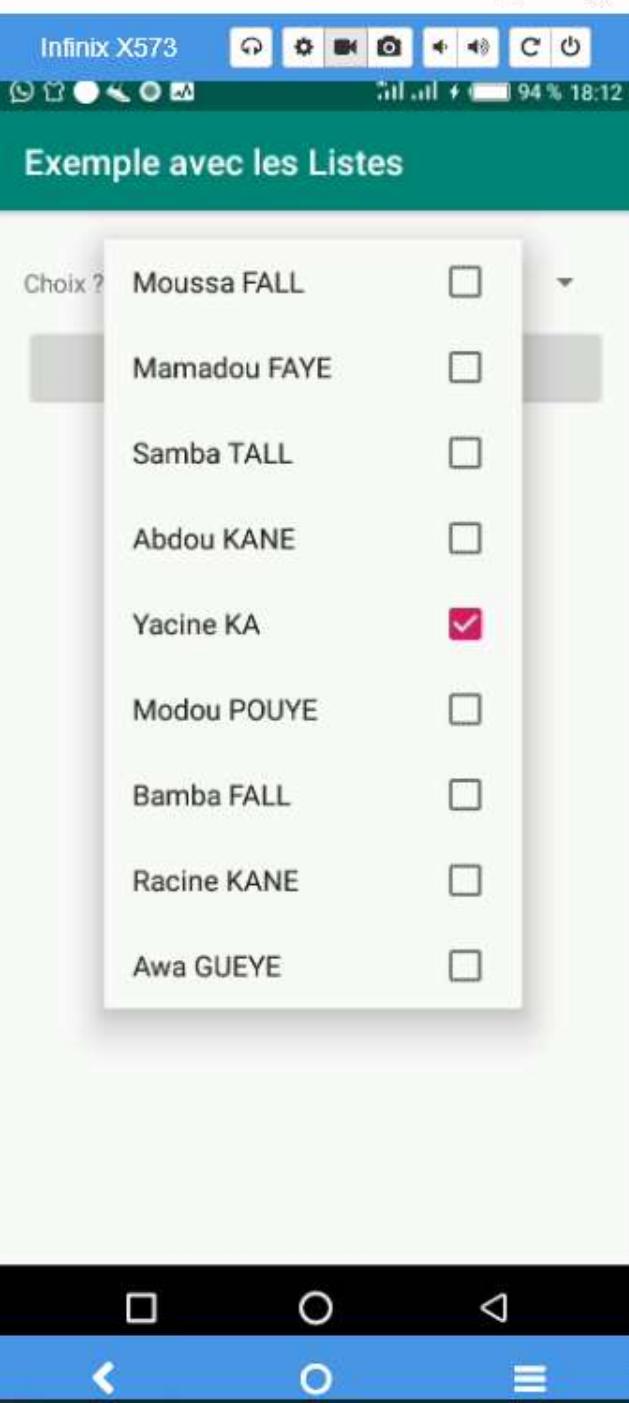
activity\_main.xml arrays.xml MainActivity.java

1: Project  
2: Favorites  
3: Build Variants  
4: Layout Captures

```

12 public class MainActivity extends AppCompatActivity {
13     Spinner liste;
14     String[] etudiants;
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19         liste = (Spinner) findViewById(R.id.listeEtudiants);
20         etudiants = getResources().getStringArray(R.array.etudiants);
21         ArrayAdapter<String> adapteur = new ArrayAdapter<String>(context: this,
22             android.R.layout.simple_list_item_multiple_choice);
23         for (int i = 0; i < etudiants.length; i++)
24             adapteur.add(etudiants[i]);
25         liste.setAdapter(adapteur);
26     }
27     public void validerChoix(View v) {
28         String choix="Vous avez choisi "+etudiants[liste.getSelectedItemPosition()];
29         Toast.makeText(context: this, choix, Toast.LENGTH_LONG).show();
30     }
31 }

```



MainActivity &gt; validerChoix()

6: Logcat TODO Terminal Build Profiler 4: Run

# GridView

ExempleaveclesListes [C:\Users\Developpeur\AndroidStudioProjects\ExempleaveclesListes] - ...\\app\\src\\main\\res\\layout\\activity\_main.xml [app] - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Project activity\_main.xml arrays.xml MainActivity.java

Preview

Palette

Gradle

Preview

Device File Explorer

23%

28px AppTheme Default (en-us)

activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="16dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Choix ?" />

    <GridView
        android:id="@+id/listeEtudiants"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:addStatesFromChildren="false"
        android:choiceMode="singleChoice"
        android:numColumns="2" />

    <Button
        android:id="@+id/button2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="validerChoix"
        android:text="Valider votre choix" />

</LinearLayout>
```

Design Text

Logcat TODO Terminal Build Profiler Run Event Log

The screenshot shows the Android Studio interface with the XML code for an activity layout. The XML defines a LinearLayout containing a TextView and a GridView. The GridView has 2 columns and 22 items. The items are labeled Item 1 through Item 22, each with a sub-item labeled Sub Item 1 through Sub Item 22. The 'Design' tab is selected, showing a preview of the layout with the GridView displaying its data. The 'Text' tab shows the raw XML code.



# FLOATING ACTION BUTTONS (FAB)

# Floating Action Buttons (FAB)

- Surélevé, circulaire, flotte au-dessus de la disposition
- Action primaire ou "promue" pour un écran
- Un par écran
- **Par exemple:** Bouton Ajouter un contact dans l'application Contacts



# Utiliser FABs

- Commencez avec le modèle d'activité de base

- **Layout:**

```
<android.support.design.widget.FloatingActionButton  
    android:id="@+id/fab"  
    android:layout_gravity="bottom|end"  
    android:layout_margin="@dimen/fab_margin"  
  
    android:src="@drawable/ic_fab_chat_button_white"  
    .../>
```

# Taille de FAB

- 56 x 56 dp par défaut
- Définissez la taille mini (30 x 40 dp) avec l'attribut **app:fabSize**:
  - `app: fabSize = "mini"`
- Réglé sur 56 x 56 dp (valeur par défaut):
  - `app: fabSize = "normal"`



1: Project

app

Gradle Scripts

Palette

Common

Text

**Buttons**

Widgets

Layouts

Containers

Google

Legacy

activity\_main.xml

Pixel 28 19% Attributes

Vous avez cliqué 0 fois sur ce TextView

Add Project Dependency

This operation requires the library com.android.support:design:+.

Would you like to add this now?

OK Cancel

Component Tree

LinearLayout(vertical)

Ab textView- "Vous avez cliqué 0 fois sur ce TextView"

Ab editText(Plain Text)

button- "Valider la saisie"

checkbox2- "Mon Exemple de checkbox"

resto- "Aller au Restaurant"

button3- "Afficher la liste"

Design Text

Build: Sync

Device File Explorer

The screenshot shows the Android Studio interface with the project 'ExempleaveclesListes' open. In the main editor, the 'activity\_main.xml' layout is displayed. A red floating action button (FAB) is present on the screen. A modal dialog titled 'Add Project Dependency' is shown, indicating that the 'com.android.support:design:+' library is required for the current operation. The 'OK' button is highlighted. The 'Component Tree' panel on the left lists several UI components: a TextView, an EditText, three buttons, and a CheckBox. The CheckBox is checked and has a tooltip 'Mon Exemple de checkbox'. The bottom status bar shows the date '04/05/2019' and time '22:14'.



1: Project

activity\_main.xml MainActivity.java

Palette

Common Text Buttons Widgets Layouts Containers Google Legacy

2: Favorites

Component Tree

LinearLayout(vertical)

- Ab textView- "Vous avez cliqué..."
- Ab editText(Plain Text)
- button- "Valider la saisie"
- checkbox2- "Mon Exemple ..."
- resto- "Aller au Restaurant"
- button3- "Afficher la liste"
- + floatingActionButton

Attributes

id	floatingActionButton
layout_width	wrap_content
layout_height	wrap_content
Layout_Margin	[?, ?, ?, ?, ?]
Padding	[?, ?, ?, ?, ?]
Theme	
elevation	
clickable	<input checked="" type="checkbox"/>
layout_gravity	[bottom, end]
onClick	chargerListe
src	@android:drawable/ic_menu_edit
adjustViewBounds	<input type="checkbox"/>
alpha	
background	
backgroundTint	<input checked="" type="checkbox"/> @android:color/holo_orange_light
backgroundTintMode	
baseline	
baselineAlignBottom	<input type="checkbox"/>
borderWidth	
contentDescription	
cropToPadding	<input type="checkbox"/>
drawingCacheQuality	

Votre nom ?

VALIDER LA SAISIE

Mon Exemple de CheckBox

ALLER AU RESTAURANT

AFFICHER LA LISTE

```
public void chargerListe(View v) {
    Intent monIntent=new Intent ( packageContext: this, ListeActivity.class);
    startActivity(monIntent);
}
```

Android Studio is ready to update.

# Gallery

- Normalement utilisé pour faire des galeries d'images avec défilement horizontal
- Propriétés supplémentaires
  - `android:animationDuration` Pour définir la durée de la transition (en ms) lorsque l'on passe d'un élément à l'autre
  - `android:unselectedAlpha` Pour définir la transparence des éléments non sélectionnés
- Pour remplir une galerie il faut un **Adapter** (comme pour ListView) mais que l'on doit écrire par héritage de la classe **BaseAdapter** puis l'associer à la galerie par la méthode `setAdapter`

Exemple de fichier XML :

```
<Gallery android:id="@+id/magalerie"  
        android:layout_width="fill_parent"  
        android:layout_height="wrap_content"  
        android:unselectedAlpha="0.5"  
    />
```



# Développement d'Applications Natives avec Android

**GESTION DE LA NAVIGATION: LES INTENTS**



# Principe des intents

- Les Intents permettent de gérer l'envoi et la réception de messages afin de faire coopérer les applications. Le but des Intents est de déléguer une action à un autre composant, une autre application ou une autre activité de l'application courante. Un objet **Intent** contient les informations suivantes:
  - le nom du composant ciblé (facultatif)
  - l'action à réaliser, sous forme de chaîne de caractères
  - les données: contenu MIME et URI
  - des données supplémentaires sous forme de paires de clef/valeur
  - une catégorie pour cibler un type d'application
  - des drapeaux (informations supplémentaires)
- On peut envoyer des Intents informatifs pour faire passer des messages. Mais on peut aussi envoyer des Intents servant à lancer une nouvelle activité

# Intents pour une nouvelle activité

- Plusieurs façons de créer l'objet de type Intent qui permettra de lancer une nouvelle activité. Si l'on passe la main à une activité interne à l'application, on peut créer l'Intent et passer la classe de l'activité ciblée par l'Intent:

```
Intent login = new Intent(this, SeLoguer.class);  
startActivity(login);
```

# Intents pour une nouvelle activité

S'il s'agit de passer la main à une autre application, on donne au constructeur de l'Intent les données et l'URI cible: l'OS est chargé de trouver une application pouvant répondre à l'Intent.

```
Button b = (Button) findViewById(R.id.Button01);
b.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        Uri telnumber = Uri.parse("tel:772051779");
        Intent call = new Intent(Intent.ACTION_CALL, telnumber);
        startActivity(call);
    }
});
```

Sans oublier la permission dans le Manifest

```
<uses-permission android:name="android.permission.CALL_PHONE" />
```

# Intent avec résultat attendu en retour

- Lorsque le bouton *retour* est pressé, l'activité courante prend fin et revient à l'activité précédente. Cela permet par exemple de terminer son appel téléphonique et de revenir à l'interface ayant initié l'appel.
- Au sein d'une application, une activité peut vouloir récupérer un code de retour de l'activité "enfant". On utilise pour cela la méthode **startActivityForResult** qui envoie un code de retour à l'activité enfant.

**public void startActivityForResult (Intent intent, int requestCode)**

- Lorsque l'activité parent reprend la main, il devient possible de filtrer le code de retour dans la méthode **onActivityResult** pour savoir si l'on revient ou pas de l'activité enfant.

**protected void onActivityResult(int requestCode, int resultCode, Intent data)**

# Retour d'une activité

- L'appel d'un *Intent* devient donc dans l'activité parent:

```
public void onCreate(Bundle savedInstanceState) {  
    Intent login = new Intent(getApplicationContext(), SaisirNumeroTelephone.class);  
    startActivityForResult(login,48);  
    ...  
}
```

- Le filtrage dans la classe parent permet de savoir qui avait appelé cette activité enfant:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == 48)  
        Toast.makeText(this, "Code de requête récupéré (je sais d'où je viens)", Toast.LENGTH_LONG).show();  
    }  
}
```

# Résultat d'une activité

- Il est aussi possible de définir un résultat d'activité, avant d'appeler explicitement la fin d'une activité avec la méthode **finish()**. Dans ce cas, la méthode **setResult** permet d'enregistrer un code de retour qu'il sera aussi possible de filtrer dans l'activité parente.
- Dans l'activité enfant, on met donc:

```
Button terminer = (Button)findViewById(R.id.terminer);
terminer.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        setResult(RESULT_OK, getIntent());
        finish();
    }
});
```

# Résultat d'une activité

- Et la classe parente peut filtrer ainsi:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data){  
    if (requestCode == 48)  
        Toast.makeText(this, "Code de requête récupéré (je sais d'où je viens)",  
                    Toast.LENGTH_LONG).show();  
    if (requestCode == 50)  
        Toast.makeText(this, "Code de retour ok (on m'a renvoyé le bon code)",  
                    Toast.LENGTH_LONG).show();  
}
```

# Ajout d'informations

- Les Intent permettent de transporter des informations à destination de l'activité cible. On appelle ces informations des Extra: les méthodes permettant de les manipuler sont **getExtra** et **putExtra**.
- Lorsqu'on prépare un Intent et que l'on souhaite ajouter une information de type
- "clef -> valeur" on procède ainsi:

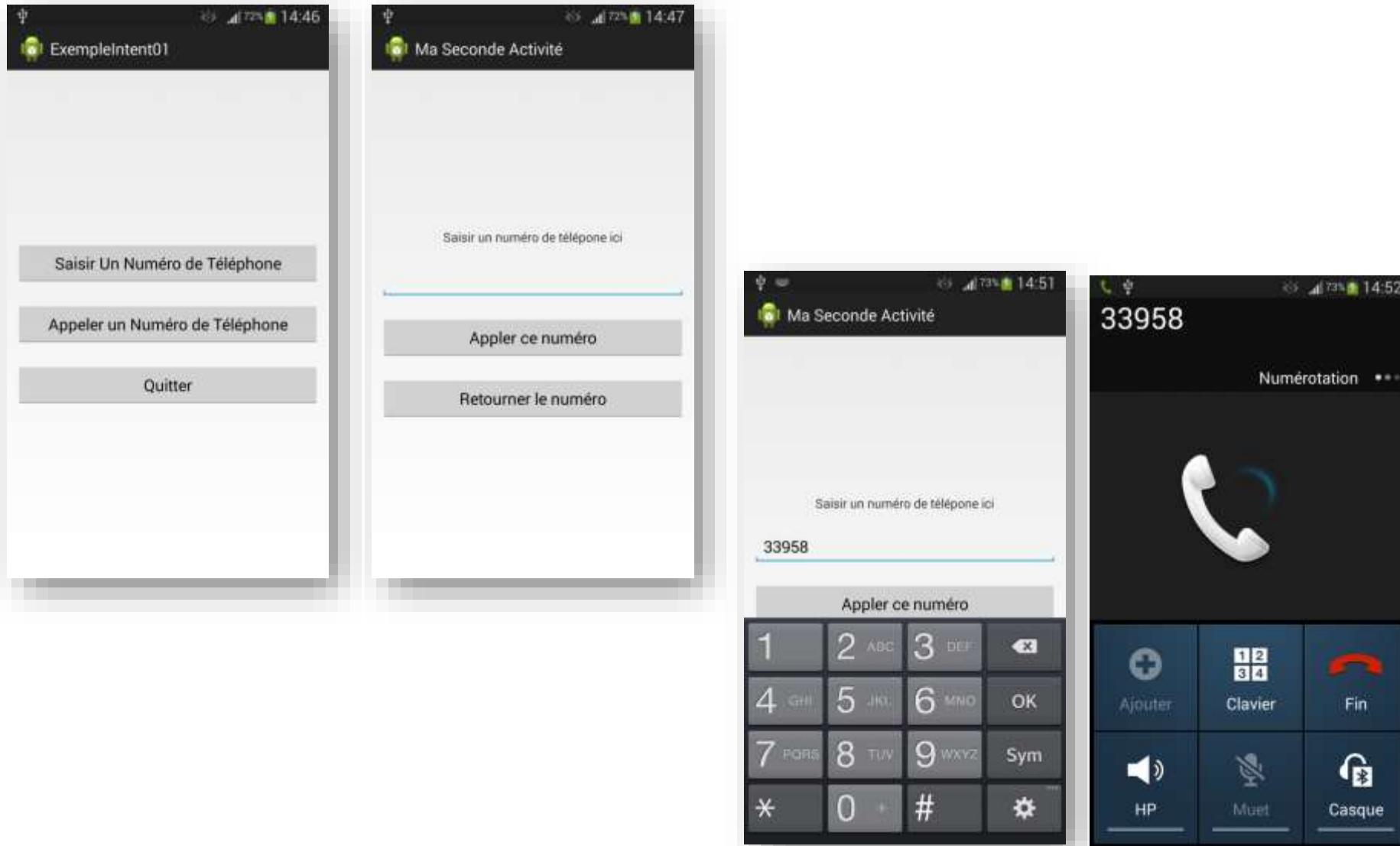
# Ajout d'informations

```
Intent callactivity2 = new Intent(getApplicationContext(), Activity2.class);
callactivity2.putExtra("login", "Lamane");
startActivity(callactivity2);
```

Du côté de l'activité recevant l'Intent, on récupère l'information de la manière suivante:

```
Bundle extras = getIntent().getExtras();
String s = new String(extras.getString("login"));
```

# Exemple



# Pour l'activité principale



# Pour l'activité principale

---

```
applerButton = (Button) findViewById(R.id.buttonAppel);
applerButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Intent appelerIntent = new Intent(getApplicationContext(),
            AppelActivity.class);
        startActivity(appelerIntent);
    }
});
numeroButton = (Button) findViewById(R.id.buttonSaisie);
numeroButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Intent saisirIntent = new Intent(getApplicationContext(),
            AppelActivity.class);
        startActivityForResult(saisirIntent, 1);
    }
});
quitterButton = (Button) findViewById(R.id.buttonQuitter);
quitterButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        finish();
    }
});
```

# Pour l'activité principale

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == 1) {
        if (resultCode == RESULT_OK) {
            String filiere = data.getStringExtra("NumTel");
            Toast.makeText(getApplicationContext(),
                "Votre numéro est " + filiere, Toast.LENGTH_LONG)
                .show();
        } else if (resultCode == RESULT_CANCELED) {
            Toast.makeText(getApplicationContext(), "Opération annulée",
                Toast.LENGTH_LONG).show();
        }
    }
}
```

# Pour la seconde activité



# Pour la seconde activité

```
numTelEditText = (EditText) findViewById(R.id.editText1);
callButton = (Button) findViewById(R.id.buttonCallNumber);
callButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Uri numTel = Uri.parse("tel:" + numTelEditText.getText());
        Intent appelerIntent = new Intent(Intent.ACTION_CALL, numTel);
        startActivity(appelerIntent);
    }
});
retourButton = (Button) findViewById(R.id.buttonBack);
retourButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        if (numTelEditText.getText().toString().trim().length() > 0) {
            getIntent().putExtra("NumTel", numTelEditText.getText().toString());
            setResult(RESULT_OK, getIntent());
        } else {
            setResult(RESULT_CANCELED, getIntent());
        }
        finish();
    }
});
```

# Intentions prédéfinies

- **ACTION\_MAIN**: action principale
- **ACTION\_VIEW**: visualiser une donnée
- **ACTION\_ATTACH\_DATA**: attachement de donnée
- **ACTION\_EDIT**: Edition de donnée
- **ACTION\_PICK**: Choisir un répertoire de donnée
- **ACTION\_CHOOSER**: menu de choix pour l'utilisateur – EXTRA\_INTENT contient l'Intent original, EXTRA\_TITLE le titre du menu
- **ACTION\_GET\_CONTENT**: obtenir un contenu suivant un type MIME
- **ACTION\_SEND**: envoyé un message (EXTRA\_TEXT|EXTRA\_STREAM) à un destinataire non spécifié

# Intentions prédéfinies

- **ACTION\_SEND\_TO**: on spécifie le destinataire dans l'URI
- **ACTION\_INSERT**: on ajoute un élément vierge dans le répertoire spécifié par l'URI
- **ACTION\_DELETE**: on supprime l'élément désigné par l'URI
- **ACTION\_PICK\_ACTIVITY**: menu de sélection selon l'EXTRA\_INTENT mais ne lance pas l'activité
- **ACTION\_SEARCH**: effectue une recherche etc...

# Intentions prédéfinies

The screenshot shows the Android Studio interface with the following components:

- Top Bar:** Shows project navigation (Nexus One), theme (AppTheme), activity (MainActivity), and build variants (Android 17).
- Layout Editor:** Displays the XML layout code for the activity. It includes a title bar with the app icon and title "ExempleIntentGallerie". Below the title bar is a vertical list of three buttons:
  - Prendre une photo avec la Camera
  - Prendre Une Image de la Galerie
  - Jouer un son MP3
- Outline View:** Located in the top right, it lists the elements in the layout:
  - LinearLayout1
    - button1 - "Prendre une photo avec la Camera"
    - button2 - "Prendre Une Image de la Galerie"
    - button3 - "Jouer un son MP3"
  - imageView1
- Properties View:** Located at the bottom right, it shows the properties for the selected element (LinearLayout1). Key properties include:
  - Id:** @+id/LinearLayout1
  - Layout Pa...**: vertical
  - Orientati...**: vertical
  - Gravity**: Gravity
  - Content ...**: LinearLay...
  - LinearLay...**: vertical
  - Orienta...**: vertical
  - Baselin...**: Baselin...
  - Weight ...**: Weight ...
  - Measur...**: Measur...
  - Divide...**: Divider
  - Show D...**: Show D...
  - Divide...**: Divider ...
  - View**: View
  - Style**: Style
  - Tag**: Tag
  - Backgr...**: Backgr...
  - Padding**: Padding
- Preview Window:** On the right side, it shows a preview of the application's user interface. The title bar says "ExempleIntentGallerie". The content area contains three buttons with the same text as the layout editor. Below the buttons is a large image of a woman with long brown hair.

# Int

```
16 public class MainActivity extends Activity {  
17     Button galerieButton, cameraButton, mp3Button;  
18     ImageView myImageView;  
19  
20     @Override  
21     protected void onCreate(Bundle savedInstanceState) {  
22         super.onCreate(savedInstanceState);  
23         setContentView(R.layout.activity_main);  
24         myImageView = (ImageView) findViewById(R.id.imageView1);  
25         cameraButton = (Button) findViewById(R.id.button1);  
26         cameraButton.setOnClickListener(new OnClickListener() {  
27             @Override  
28             public void onClick(View arg0) {  
29                 Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
30                 startActivityForResult(intent, 1);  
31             }  
32         });  
33         galerieButton = (Button) findViewById(R.id.button2);  
34         galerieButton.setOnClickListener(new OnClickListener() {  
35             @Override  
36             public void onClick(View arg0) {  
37                 startActivityForResult(  
38                     new Intent(Intent.ACTION_PICK,  
39                         android.provider.MediaStore.Images.Media.INTERNAL_CONTENT_URI),2);  
40             }  
41         });  
42         mp3Button = (Button) findViewById(R.id.button3);  
43         mp3Button.setOnClickListener(new OnClickListener() {  
44             @Override  
45             public void onClick(View arg0) {  
46                 Intent intent = new Intent();  
47                 intent.setType("audio/mp3");  
48                 intent.setAction(Intent.ACTION_GET_CONTENT);  
49                 startActivityForResult(Intent.createChooser(intent, "Open Audio (mp3) file"),3);  
50             }  
51         });  
52     }  
}
```

```
Int 54    public void onActivityResult(int requestCode, int resultCode, Intent intent) {  
55        super.onActivityResult(requestCode, resultCode, intent);  
56        switch (requestCode) {  
57            case 1:  
58                if (resultCode == Activity.RESULT_OK) {  
59                    Bitmap bitmapImage = (Bitmap) intent.getExtras().get("data");  
60                    myImageView.setImageBitmap(bitmapImage);  
61                    myImageView.setVisibility(ImageView.VISIBLE);  
62                }  
63                break;  
64  
65            case 2:  
66                if (resultCode == Activity.RESULT_OK) {  
67                    Uri mImageCaptureUri = intent.getData();  
68                    myImageView.setImageURI(mImageCaptureUri);  
69                    myImageView.setVisibility(ImageView.VISIBLE);  
70                }  
71                break;  
72  
73            case 3:  
74                if (resultCode == Activity.RESULT_OK) {  
75                    Uri audioFileUri = intent.getData();  
76                    MediaPlayer myMediaPlayer = MediaPlayer.create(this, audioFileUri);  
77                    myMediaPlayer.start();  
78  
79                    // info.setText(audioFileUri.getPath());  
80                }  
81                break;  
82            }  
83        }  
84    }
```

# Développement d'Applications Natives avec Android

**ANDROID- ÉLÉMENTS STYLE:  
POLICES DE CARACTÈRES, STYLES ET THÈMES**

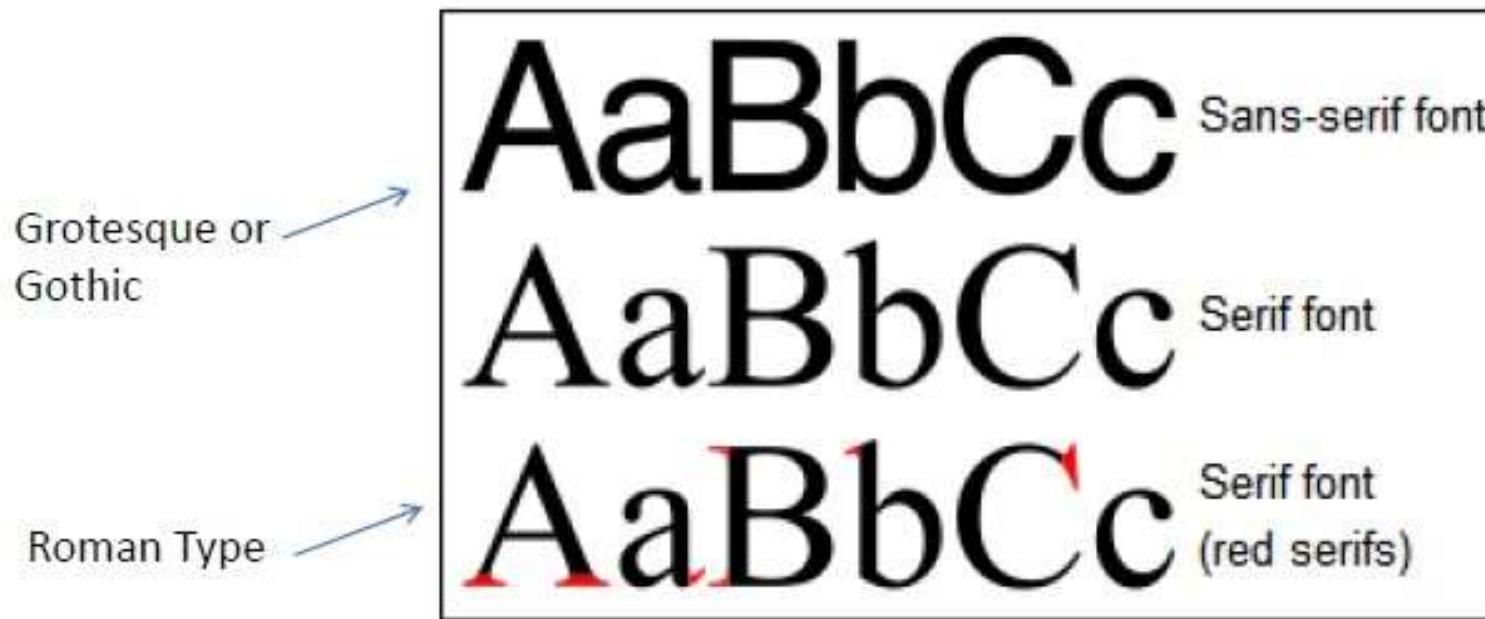


# *LES FONTS*

# Les Fonts

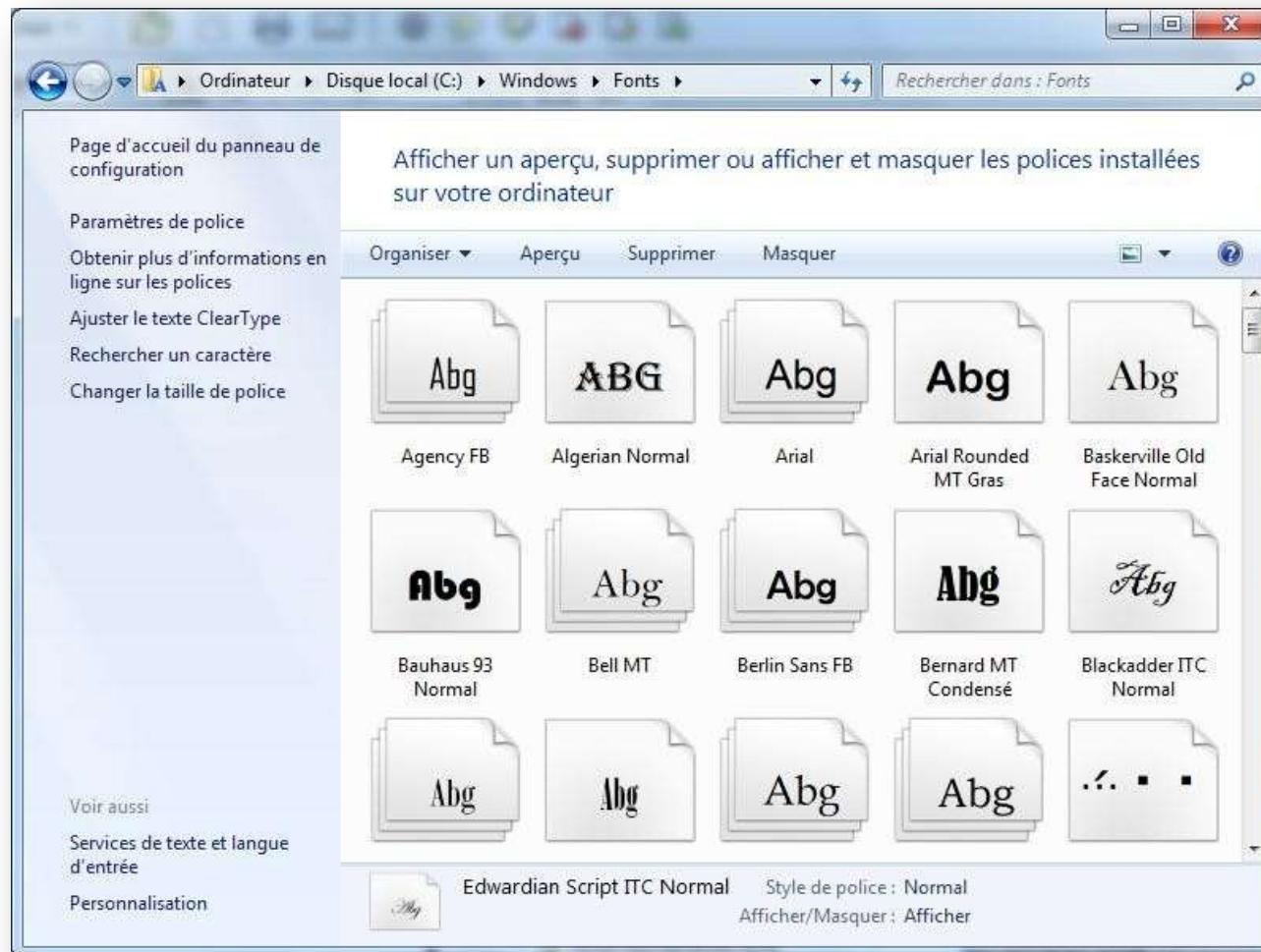
- Android propose naturellement trois familles de fonts:
  - **sans**: Une étudiante fantôme en LGI
  - **serif**: Une étudiante fantôme en LGI
  - **monospaced**: Une étudiante fantôme

# Fonts



Source: <http://en.wikipedia.org/wiki/Serif>

# Les autres Fonts(Windows)



The screenshot shows the Android Studio interface with the project 'ExempleaveclesListes' open. The 'activity\_main.xml' file is selected in the top navigation bar.

A context menu is open over the 'res' folder in the Project tool window. The 'New' option is highlighted, and a submenu is displayed:

- Kotlin File/Class
- Android Resource File
- Android Resource Directory**
- Sample Data Directory
- File
- Scratch File
- Directory
- C++ Class
- C/C++ Source File
- C/C++ Header File
- Image Asset
- Vector Asset
- Kotlin Script
- Singleton
- Gradle Kotlin DSL Build Script
- Gradle Kotlin DSL Settings
- Edit File Templates...

To the right of the submenu, a 'New Resource Directory' dialog is open. The 'Directory name:' field contains 'font'. The 'Resource type:' dropdown is set to 'font'. The 'Source set:' dropdown is set to 'main'. The 'Available qualifiers:' section lists various qualifiers, and the 'Chosen qualifiers:' section is currently empty. At the bottom of the dialog are 'OK', 'Cancel', and 'Help' buttons.

The Project tool window on the left shows the project structure with 'app' as the root module. The 'res' folder is expanded, showing its contents. Other tool windows like 'Gradle', '2: Favorites', 'Build Variants', 'Layout Captures', and 'Device File Explorer' are visible along the left edge.

1: Project

```
<?xml version="1.0" encoding="utf-8?
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="16dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Choix ?" />

    <GridView
        android:id="@+id/listeEtudia
        android:layout_width="match_j
        android:layout_height="wrap_
        android:addStatesFromChildre
        android:choiceMode="singleCh
        android:numColumns="2" />

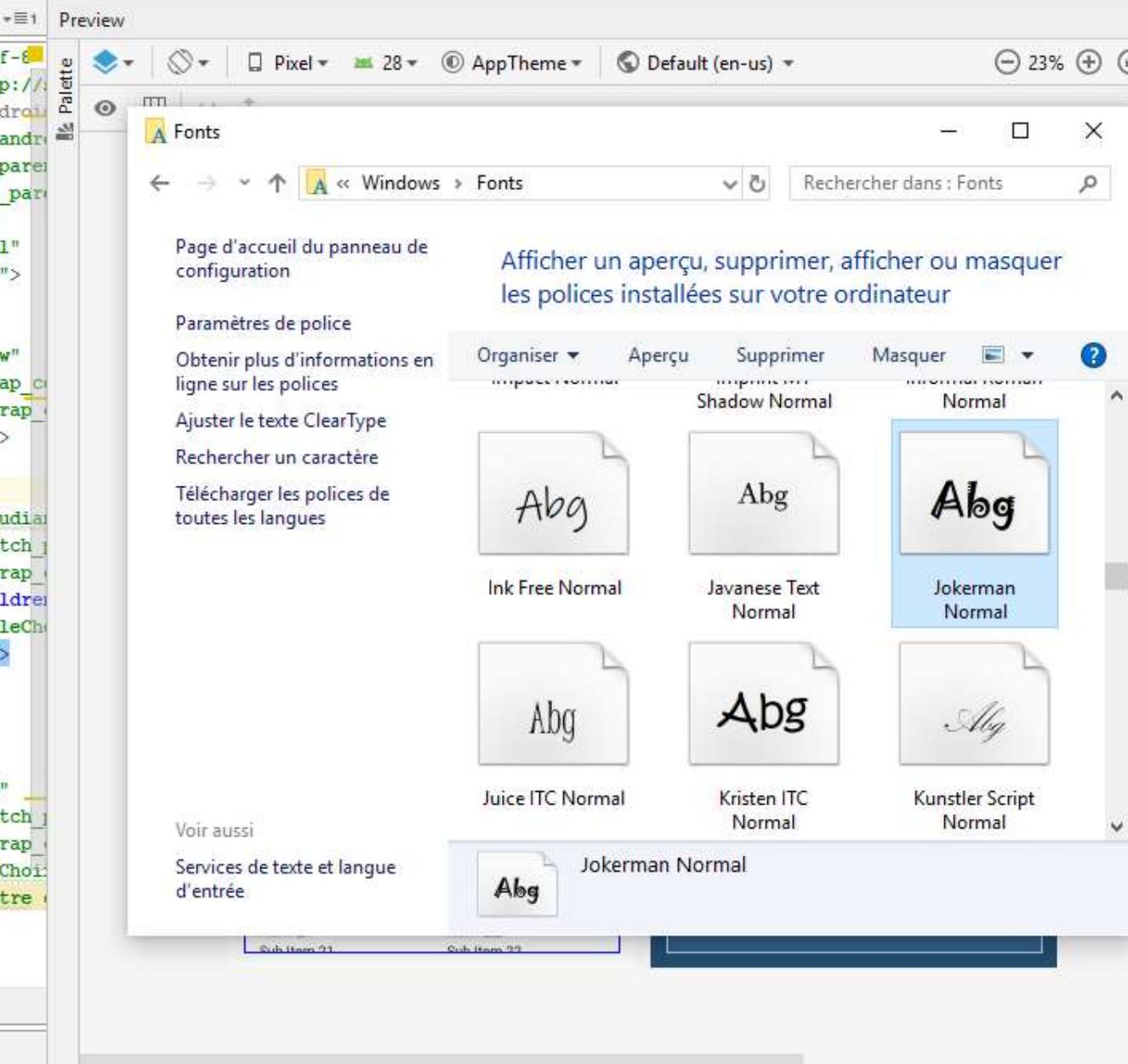
    <Button
        android:id="@+id/button2"
        android:layout_width="match_j
        android:layout_height="wrap_
        android:onClick="validerChois
        android:text="Valider votre" />
</LinearLayout>
```

2: Favorites

3: Build Variants

4: Layout Captures

5: Device File Explorer





1: Project

2: Structure

3: Favorites

4: Build Variants

5: Layout Captures

6: Device File Explorer

7: Event Log

activity\_main.xml arrays.xml MainActivity.java

```

1 package com.example.exempleavecleslistes;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6     GridView liste;
7     String[] etudiants;
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13        liste = (GridView) findViewById(R.id.gridView1);
14
15        Button b=(Button)findViewById(R.id.button1);
16
17        etudiants = getResources().getStringArray(R.array.array);
18        ArrayAdapter<String> adapteur = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, etudiants);
19        for (int i = 0; i < etudiants.length; i++) {
20            adapteur.add(etudiants[i]);
21        }
22        liste.setAdapter(adapteur);
23
24    }
25
26    public void validerChoix(View v) {
27        String choix="Vous avez choisi "+etudiants[0];
28        Toast.makeText(context, choix, Toast.LENGTH_SHORT).show();
29    }
30
31 }
32
33
34
35
36

```

**Fonts**

Page d'accueil du panneau de configuration

Paramètres de police

Obtenir plus d'informations en ligne sur les polices

Ajuster le texte ClearType

Rechercher un caractère

Télécharger les polices de toutes les langues

Voir aussi

Services de texte et langue d'entrée

Organiser Aperçu Supprimer Masquer

Shadow Normal Normal

Ink Free Normal Javanese Text Normal Jokerman Normal

Juice ITC Normal Kristen ITC Normal Kunstler Script Normal

Jokerman Normal

MainActivity > onCreate()

6: Logcat 7: TODO 8: Terminal 9: Build 10: Profiler 11: Run 12: Event Log

1: Project

2: Favorites

3: Build Variants

4: Layout Captures

5: Device File Explorer

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

activity\_main.xml arrays.xml MainActivity.java jokerman.ttf

Android

app

- manifests
- java
  - com.example.exempleavecleslistes
    - MainActivity
- generatedJava
- res
  - drawable
  - font
    - jokerman.ttf
  - layout
  - mipmap
  - values
    - arrays.xml
    - colors.xml
    - strings.xml
    - styles.xml

- Gradle Scripts

Jokerman

**Lorem ipsum dolor sit amet, consectetur adipisicing elit. Ad aut cum,  
est exercitationem explicabo facilis illum itaque iure, iusto possimus  
quae qui quod quos sed, velit? Exercitationem harum inventore ipsa  
magnam magni quae quos. Alias cum dolor eveniet fuga molestiae  
molestias odit rerum voluptatum. Adipisci placeat quis saepe ullam  
voluptates.**

6: Logcat   7: TODO   8: Terminal   9: Build   10: Profiler   11: Run   12: Event Log

1: Project

2: Favorites

3: Build Variants

4: Layout Captures

5: Device File Explorer

6: Logcat

7: TODO

8: Terminal

9: Build

10: Profiler

11: Run

12: Event Log

13: Instant Run applied code changes and restarted the app. R Class Change. // (Don't show again) (2 minutes ago)

14: Context: <no context>

15: Smiley

16: Sad

17: Neutral

18: Project

19: activity\_main.xml

20: arrays.xml

21: MainActivity.java

22: jokerman.ttf

23: Attributes

24: Palettes

25: Common

26: Text

27: Buttons

28: Widgets

29: Layouts

30: Containers

31: Google

32: Legacy

33: Component Tree

34: LinearLayout(vertical)

35: textView- "Choix ?"

36: listeEtudiants

37: button2- "Valider vot..."

38: Design

39: Text

40: Pixel

41: 28

42: AppTheme

43: 100%

44: Choix ?

45: Item 1

46: Sub Item 1

47: Item 3

48: Sub Item 3

49: fontFamily @font/jokerman

50: foreground

51: foregroundGravity []

52: freezesText

53: gravity []

54: hapticFeedbackEnab

55: height

56: hint

57: imeActionId

58: imeActionLabel

59: imeOptions []

60: includeFontPadding

61: inputMethod

62: inputType []

63: isScrollContainer



1: Project

2: Structure

3: Favorites

4: Build Variants

5: Layout Captures

activity\_main.xml arrays.xml MainActivity.java jokerman.ttf

```

1 package com.example.exempleavecleslistes;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6     GridView liste;
7     String[] etudiants;
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12         liste = (GridView) findViewById(R.id.ListeEtudiants);
13
14         Button b=(Button)findViewById(R.id.button2);
15
16         Typeface typeface = ResourcesCompat.getFont(context, R.font.jokerman);
17         b.setTypeface(typeface);
18
19         etudiants = getResources().getStringArray(R.array.etudiants);
20         ArrayAdapter<String> adapteur = new ArrayAdapter<String>(context, android.R.layout.simple_list_item_single_choice);
21         for (int i = 0; i < etudiants.length; i++)
22             adapteur.add(etudiants[i]);
23         liste.setAdapter(adapteur);
24     }
25
26     public void validerChoix(View v) {
27         String choix="Vous avez choisi "+etudiants[liste.getSelectedItemPosition()];
28         Toast.makeText(context, choix, Toast.LENGTH_LONG).show();
29     }
30
31 }
32
33
34
35
36
37
38
39

```

MainActivity &gt; onCreate()



# STYLES

# THEMES

# Développement d'Applications Natives avec Android

**ANDROID- ÉLÉMENTS D'INTERACTION:  
BOÎTES DE DIALOGUE, NOTIFICATIONS, MENUS,  
SÉLECTION DATES ET HEURES**



# Boite de dialogue: La classe Toast

- Texte qui apparaît en premier plan puis disparaît au bout d'un temps donné
- Création d'un Toast
  - `Toast.makeText(Context, String, int)` renvoie l'objet de classe Toast créé.
  - Le premier paramètre est l'activité
  - Le deuxième paramètre est le message à afficher
  - Le dernier paramètre indique la durée d'affichage les seules valeurs possibles sont : `Toast.LENGTH_SHORT` (2 secondes) ou `Toast.LENGTH_LONG` (5 secondes).
- Positionnement d'un Toast
  - `setGravity(int, int, int)` appelée avant l'affichage par `show` pour indiquer où s'affichera le message.
  - Le premier paramètre sert à placer le message par rapport à l'écran. Il peut prendre l'une des valeurs définies dans la classe Gravity soit : `Gravity`. (`TOP`, `BOTTOM`, `LEFT`, `RIGHT`, `CENTER_VERTICAL`, `FILL_VERTICAL`, `CENTER_HORIZONTAL`, `FILL_HORIZONTAL`, `CENTER`, `FILL`).
  - Les deux paramètres suivants indiquent le décalage (en pixels).
- Affichage d'un Toast
  - `show()` affiche le message pour la durée définie lors de sa création.

Répertoire illisible

# Boite de dialogue: Exemple de Toast

- Un Toast est un message qui apparaît et disparaît
- Il est impossible de savoir si l'utilisateur l'a vu ou pas
- Configuration:
  - Un String contenant le message
  - Une durée
    - Toast.LENGTH\_LONG ou Toast.LENGTH\_SHORT
- Il est aussi possible de donner une View de votre choix en paramètre



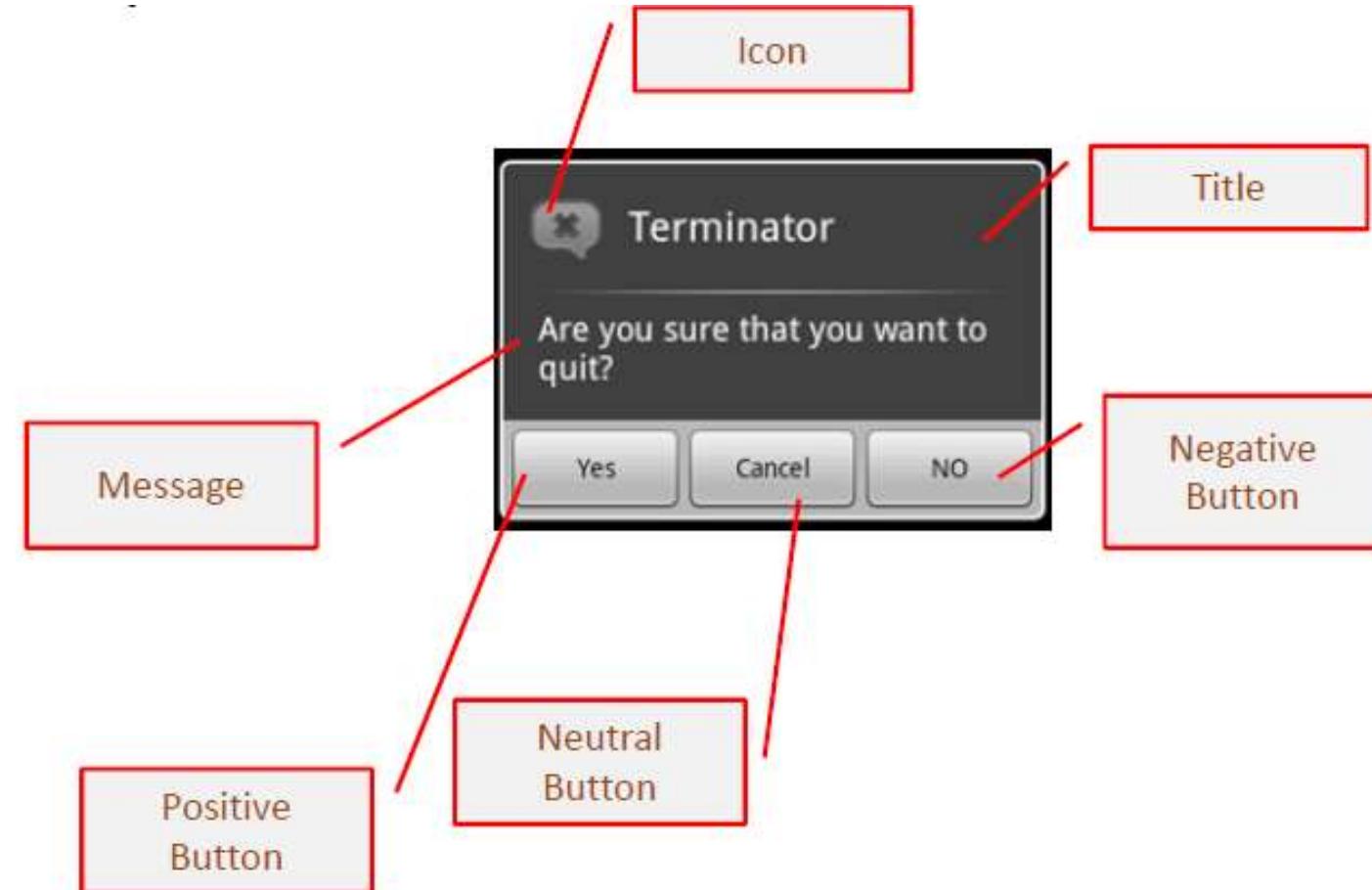
# Toast : exemple

```
Toast.makeText(MainActivity.this,  
        "C'est vous qui voyez !!!",  
        Toast.LENGTH_SHORT).show();
```

# Boîte de dialogue: La classe AlertDialog

- Style plus classique des boîtes de dialogue. Un AlertDialog s'ouvre, prend le focus et reste affiché tant que l'utilisateur ne le ferme pas
- Pour créer un AlertDialog:
  - Utiliser la classe Builder offrant un ensemble de méthodes permettant de configurer un AlertDialog. Méthodes renvoie le Builder afin de faciliter le chaînage des appels.
  - À la fin, il suffit d'appeler la méthode show() de l'objet Builder pour afficher la boîte de dialogue.
- Méthodes de configuration de Builder :
  - setMessage() permet de définir le "corps" de la boîte de dialogue
  - setTitle() et setIcon() permettent de configurer le texte et/ou l'icône
  - setPositiveButton(), setNeutralButton() et setNegativeButton() permettent d'indiquer les boutons qui apparaîtront en bas de la boîte de dialogue, leur emplacement latéral (respectivement, à gauche, au centre ou à droite), leur texte et le code qui sera appelé lorsqu'on clique sur un bouton (en plus de refermer la boîte de dialogue).

# Boîte de dialogue: La classe AlertDialog



1: Project

MainActivity.java activity\_main.xml exit.png macky\_sall.jpg

Palette

Common Text Buttons Widgets Layouts Containers Google Legacy

2: Favorites

Component Tree

LinearLayout(vertical)

- Ab textView- "Vous avez cliqué..."
- Ab editText(Plain Text)
- button- "Valider la saisie"
- checkbox2- "Mon Exemple ..."
- resto- "Aller au Restaurant"
- button3- "Afficher la liste"
- button2- "Quitter"
- + floatingActionButton

Design Text

6: Logcat TODO Terminal Build Profiler 4: Run

Context: <no context>

Attributes

- nextFocusDown
- nextFocusForward
- nextFocusLeft
- nextFocusRight
- nextFocusUp
- ▶ numeric
- onClick
- overScrollMode
- password
- phoneNumber
- privateimeOptions
- ▶ requiresFadingEdge
- rotation
- rotationX
- rotationY
- saveEnabled
- scaleX
- scaleY
- scrollHorizontally
- scrollX
- scrollY
- scrollbarAlwaysDrawHorizontalTrack
- scrollbarAlwaysDrawVerticalTrack
- scrollbarDefaultDelayBeforeFade
- scrollbarFadeDuration
- scrollbarSize
- scrollbarStyle
- scrollbarThumb
- scrollbarThumb

IDE and Plugin Updates  
Android Studio is ready to update.

Grade Device File Explorer



1: Project

2: Structure

2: Favorites

Build Variants

Layout Captures

```

64     public void quitter(View v) {
65         // Créer l'alert builder
66         AlertDialog.Builder builder = new AlertDialog.Builder( context: this );
67         builder.setTitle("Question");
68         builder.setMessage("Voulez-vous vraiment quitter ?");
69         // Ajouter les bouttons
70         builder.setPositiveButton( text: "Oui", new DialogInterface.OnClickListener() {
71             @Override
72             public void onClick(DialogInterface dialog, int which) {
73                 finish();
74             }
75         });
76         builder.setNeutralButton( text: "Non", listener: null );
77         builder.setNegativeButton( text: "Abandonner", listener: null );
78         // créer et montrer la boîte alerte
79         AlertDialog dialog = builder.create();
80         dialog.setIcon(R.drawable.macky_sall);
81         dialog.show();
82     }
83 }
84

```





1: Project

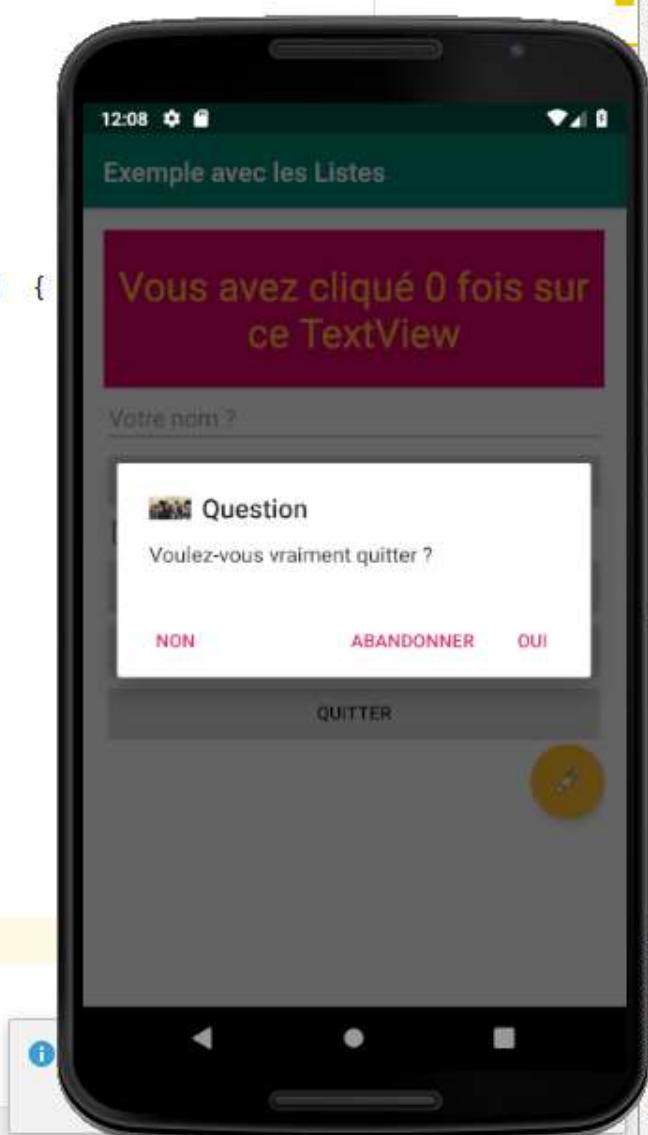
```
64     public void quitter(View v) {
65         // Créer l'alert builder
66         AlertDialog.Builder builder = new AlertDialog.Builder( context: this );
67         builder.setTitle("Question");
68         builder.setMessage("Voulez-vous vraiment quitter ?");
69         // Ajouter les bouttons
70         builder.setPositiveButton( text: "Oui", new DialogInterface.OnClickListener() {
71             @Override
72             public void onClick(DialogInterface dialog, int which) {
73                 finish();
74             }
75         });
76         builder.setNeutralButton( text: "Non", listener: null );
77         builder.setNegativeButton( text: "Abandonner", listener: null );
78         // créer et montrer la boîte alerte
79         AlertDialog dialog = builder.create();
80         dialog.setIcon(R.drawable.macky_sall);
81         dialog.show();
82     }
83 }
84 }
```

2: Structure

3: Favorites

4: Build Variants

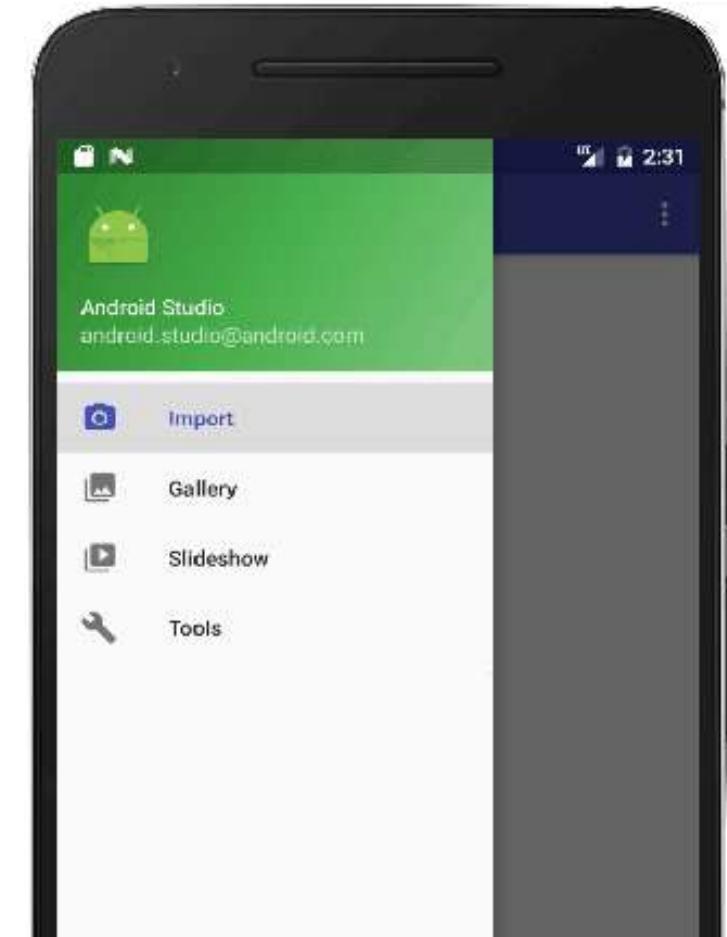
5: Layout Captures



# NAVIGATION DRAWER

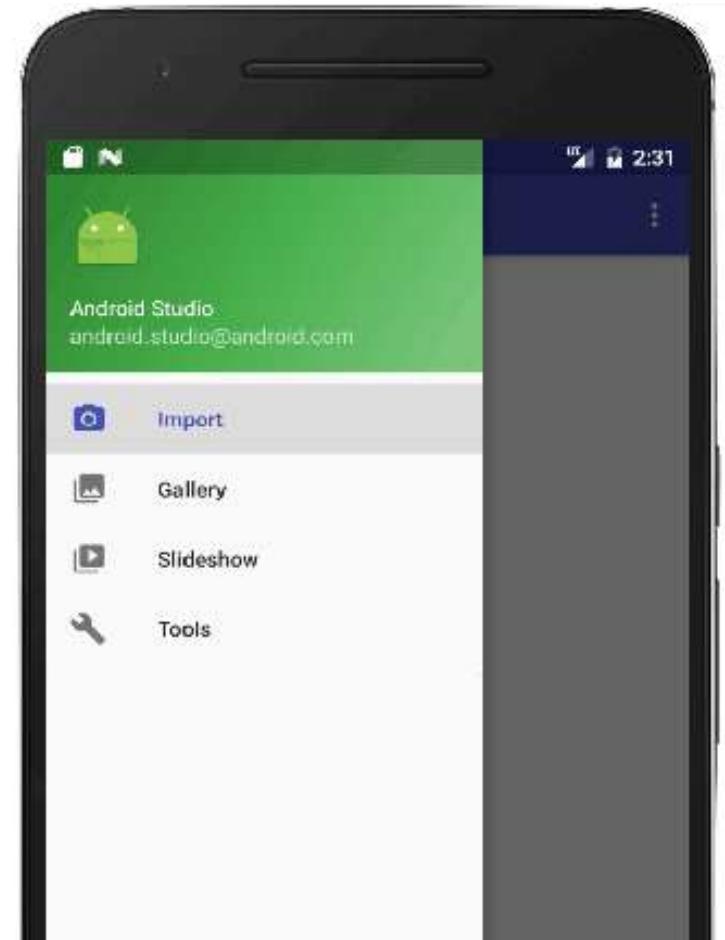
# Navigation Drawer

- The navigation drawer est un panneau d'interface utilisateur qui affiche le menu de navigation principal de votre application.
- Il est masqué lorsqu'il n'est pas utilisé, mais apparaît lorsque l'utilisateur fait glisser un doigt du bord gauche de l'écran.



# Drawer Layout

- Ajouter à Android v 22.1
  - Donc, vous devez utiliser la bibliothèque de support Android
- Déclarez votre mise en page racine avec DrawerLayout.
- Dans DrawerLayout, ajoutez une disposition pour le contenu principal de l'interface utilisateur et une autre vue contenant le contenu du navigation drawer.



# Drawer Layout

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Use DrawerLayout as root container for activity -->
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

</android.support.v4.widget.DrawerLayout>
```

# Drawer Layout

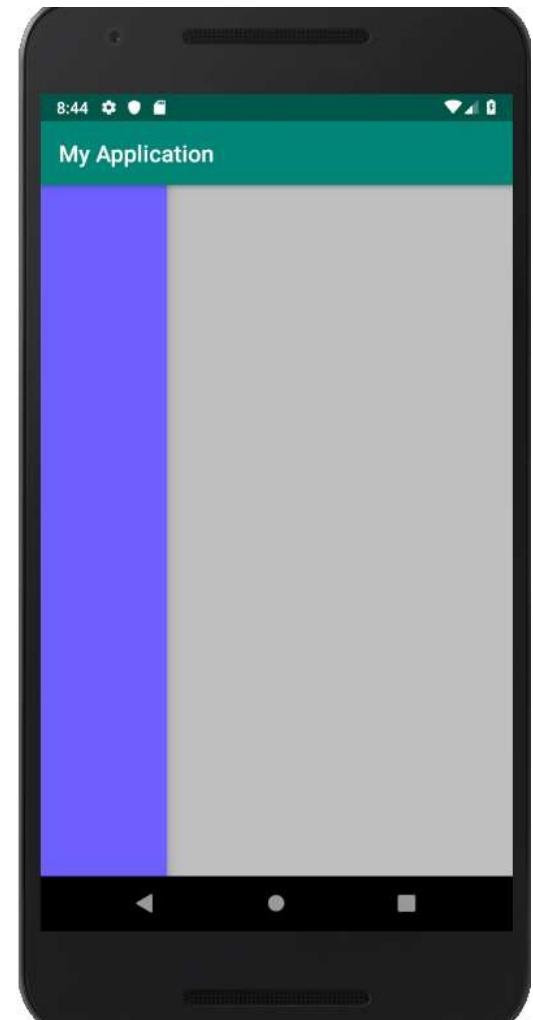
```
<?xml version="1.0" encoding="utf-8"?>
<!-- Use DrawerLayout as root container for activity -->
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <!-- Layout to contain contents of main body -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"></LinearLayout>

</android.support.v4.widget.DrawerLayout>
```

# Drawer Layout

```
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">
    <!-- Layout to contain contents of main body -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"></LinearLayout>
    <!-- Container for contents of drawer -->
    <android.support.design.widget.NavigationView
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"/>
</android.support.v4.widget.DrawerLayout>
```



# Items de menu pour nav drawer

Créer une ressource de menu avec le nom de fichier correspondant

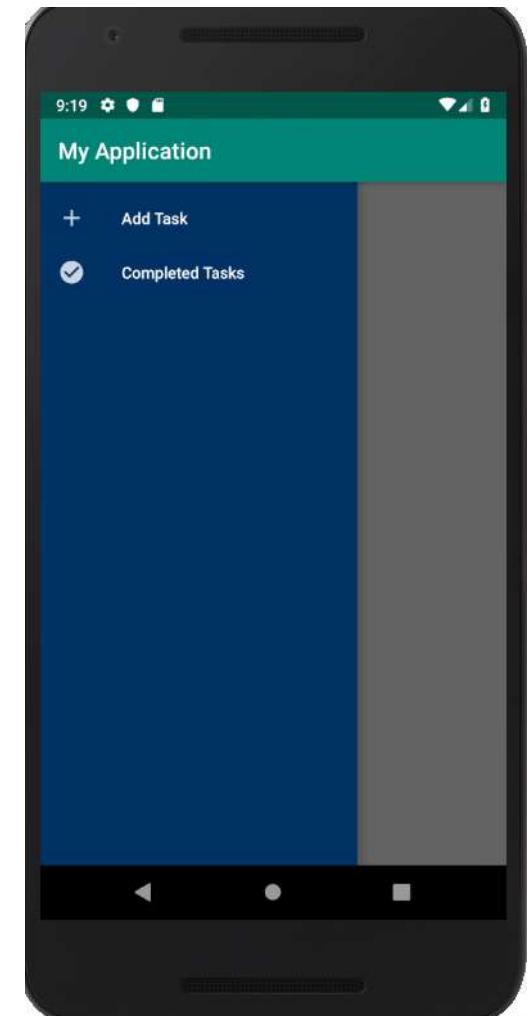
Ajouter des éléments de menu au fichier de ressources

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group>
        <item
            android:id="@+id/nav_camera"
            android:icon="@drawable/ic_addw"
            android:title="Add Task" />
        <item
            android:id="@+id/nav_gallery"
            android:icon="@drawable/ic_list"
            android:title="Completed Tasks" />
    </group>
</menu>
```

# Menu items for nav drawer

- Configurez les éléments de menu répertoriés dans le nav drawer
- Spécifiez une ressource de menu avec l'attribut app: menu

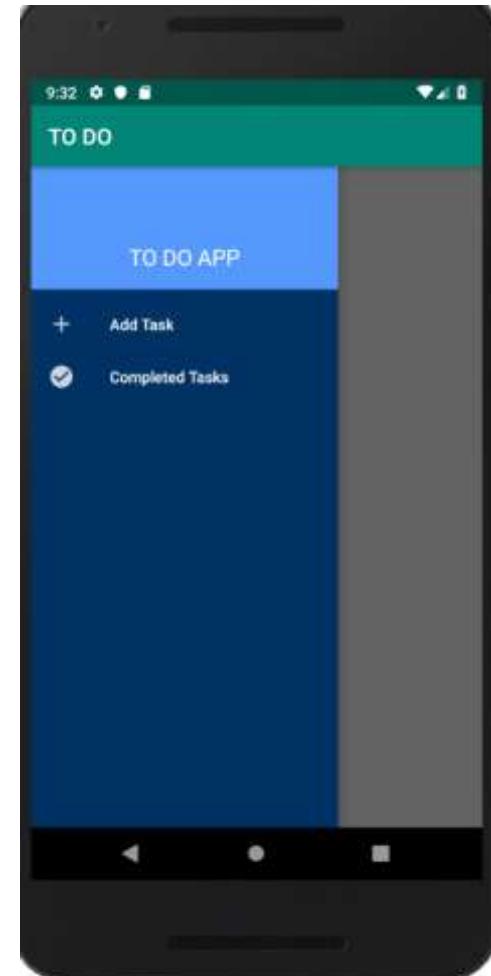
```
<android.support.design.widget.NavigationView  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    android:background="#003366"  
    app:itemIconTint="@android:color/white"  
    app:itemTextColor="@android:color/white"  
    app:menu="@menu/navigation_menu" />
```



# Ajouter un en-tête à la nav drawer

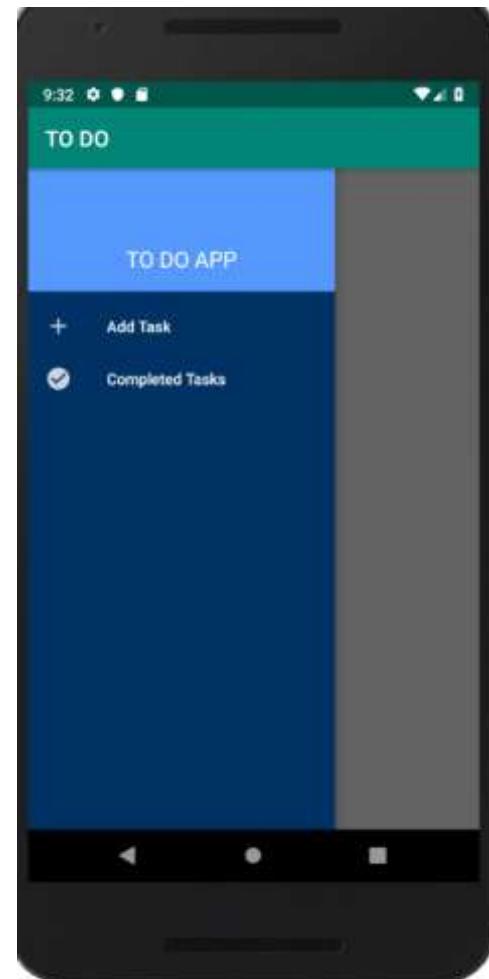
- Vous pouvez ajouter un en-tête au nv drawer en spécifiant une présentation avec l'attribut **app: headerLayout**.

```
<android.support.design.widget.NavigationView  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    android:background="#003366"  
    app:itemIconTint="@android:color/white"  
    app:itemTextColor="@android:color/white"  
    app:menu="@menu/navigation_menu"  
    app:headerLayout="@layout/nav_header"/>
```



# Ajouter l'entête

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="192dp"
    android:background="#5599ff"
    android:gravity="bottom"
    android:orientation="vertical"
    android:padding="16dp"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="TO DO APP"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
        android:textSize="20sp"
        android:textStyle="bold" />
</LinearLayout>
```



# Gestion de la navigation au click

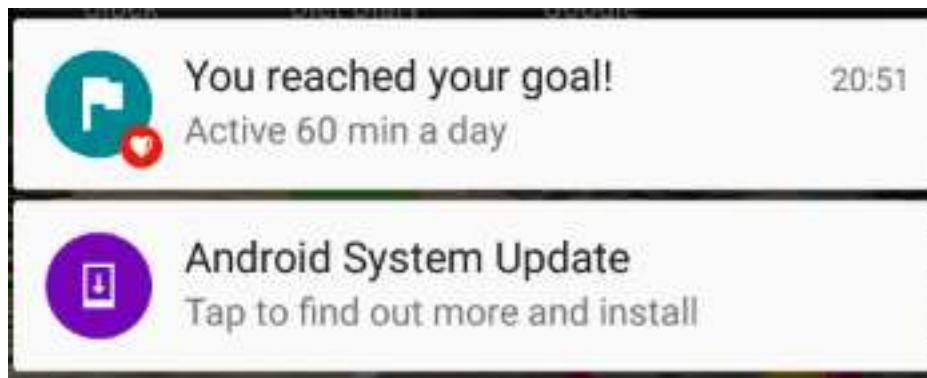
Pour recevoir des rappels, implémentez l'interface OnNavigationItemSelectedListener et ajoutez-la à votre NavigationView en appelant setNavigationItemSelectedListener () .

```
NavigationView navigationView = findViewById(R.id.nav_view);
navigationView.setNavigationItemSelectedListener(
    new NavigationView.OnNavigationItemSelectedListener() {
        @Override
        public boolean onNavigationItemSelected(MenuItem menuItem){
            if (menuItem.getTitle().toString().startsWith("Add")){
                getJob();
            }
            mDrawerLayout.closeDrawers();
            return true;
        }
    });
});
```

# NOTIFICATIONS

# Qu'est-ce qu'une notification?

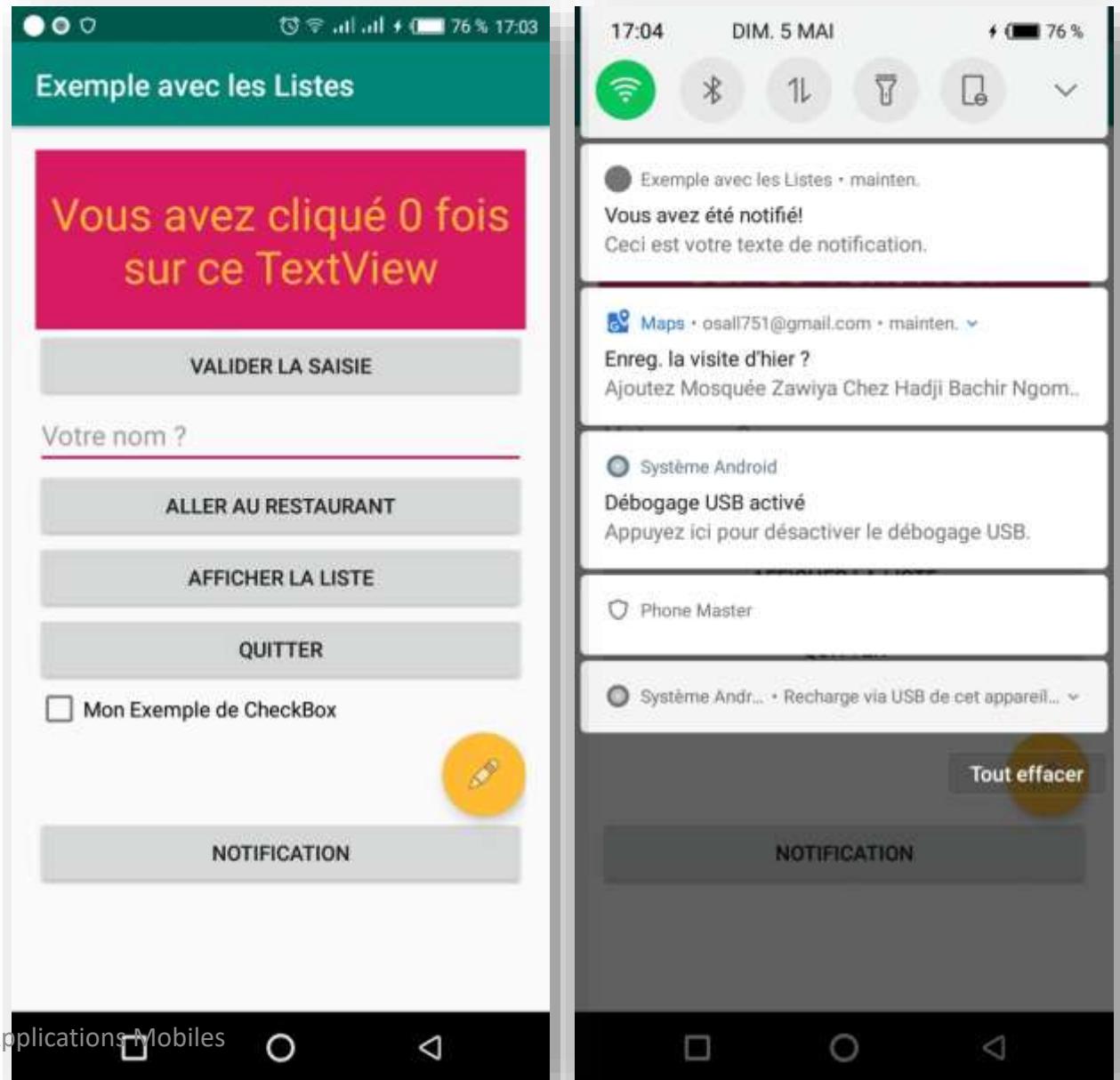
Message affiché à l'utilisateur en dehors de l'interface utilisateur de l'application normale



- Small icon
- Title
- Detail text

# Comment les notifications sont-elles utilisées?

- Android émet une notification qui apparaît sous forme d'icône dans la barre d'état.
- Pour voir les détails, l'utilisateur ouvre le tiroir de notification.
- L'utilisateur peut afficher les notifications à tout moment dans le tiroir de notifications.



# Notification channels

- Utilisé pour créer un canal personnalisable par l'utilisateur pour chaque type de notification à afficher.
- Plusieurs notifications peuvent être regroupées dans un canal.
- Définissez le comportement de notification comme son, lumière, vibration, etc., appliqué à toutes les notifications de ce canal.

# Les canaux de notification sont obligatoires

- Les canaux de notification sont introduits dans Android 8.0 (API niveau 26)
- Toutes les notifications doivent être attribuées à un canal à partir d'Android 8.0 (API niveau 26), sinon vos notifications ne seront pas affichées.
- Pour les applications ciblant une version inférieure à Android 8.0 (API niveau 26), nul besoin d'implémenter des canaux de notification.

# Créer un canal de notification

Vous devez spécifier:

- Un identifiant unique dans votre package.
- Nom visible de l'utilisateur du canal.
- Le niveau d'importance pour le canal.

```
String CHANNEL_ID="ID de mon Channel";
NotificationManager mNotifyManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    NotificationChannel notificationChannel = new NotificationChannel(CHANNEL_ID,
        "Mascot Notification",
        NotificationManager.IMPORTANCE_DEFAULT);
    notificationChannel.enableLights(true);
    notificationChannel.setLightColor(Color.RED);
    notificationChannel.enableVibration(true);
    notificationChannel.setDescription("Notification du Mascot");
    mNotifyManager.createNotificationChannel(notificationChannel);
}
```

# Importance level

- Disponible dans Android 8.0 (API niveau 26) et supérieur.
- Définit le niveau d'intrusion, comme le son et la visibilité de toutes les notifications publiées sur le canal.
- Plage comprise entre IMPORTANCE\_NONE (0) et IMPORTANCE\_HIGH (4).
- Pour prendre en charge les versions antérieures d'Android (niveau inférieur à l'API 26), définissez la priorité.

# Importance level and priority constants

User-visible importance level	Importance (Android 8.0 and higher)	Priority (Android 7.1 and lower)
<b>Urgent</b> Makes a sound and appears as a heads-up notification	<a href="#"><u>IMPORTANCE_HIGH</u></a>	<a href="#"><u>PRIORITY_HIGH</u></a> or <a href="#"><u>PRIORITY_MAX</u></a>
<b>High</b> Makes a sound	<a href="#"><u>IMPORTANCE_DEFAULT</u></a>	<a href="#"><u>PRIORITY_DEFAULT</u></a>
<b>Medium</b> No sound	<a href="#"><u>IMPORTANCE_LOW</u></a>	<a href="#"><u>PRIORITY_LOW</u></a>
<b>Low</b> No sound and doesn't appear in the status bar	<a href="#"><u>IMPORTANCE_MIN</u></a>	<a href="#"><u>PRIORITY_MIN</u></a>

# Creating Notification

- La notification est créée à l'aide de la classe **NotificationCompat.Builder**.
- Transmettez le contexte de l'application et l'ID du canal de notification au constructeur.
- Le constructeur **NotificationCompat.Builder** utilise l'ID du canal de notification. Ce dernier n'est utilisé que par Android 8.0 (API de niveau 26) et supérieur, mais ce paramètre est ignoré par les versions antérieures.

# Définition du contenu de la notification

- Une petite icône, définie par `setSmallIcon ()`.
- C'est le seul contenu requis.
- Un titre, défini par `setContentTitle ()`.
- Le corps du texte, défini par `setContentText ()`.  
Ceci est le message de notification.



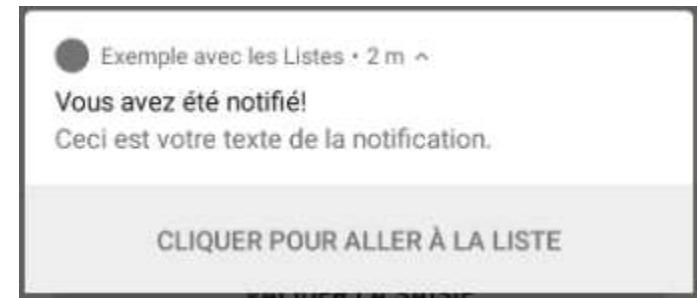
# Définition du contenu de la notification

```
NotificationCompat.Builder mBuilder =new NotificationCompat.Builder( context: this, CHANNEL_ID)
    .setSmallIcon(R.mipmap.ic_launcher_round)
    .setContentTitle("Vous avez été notifié!")
    .setContentText("Ceci est votre texte de notification.");
mNotifyManager.notify( id: 1, mBuilder.build());
```

# Ajouter une notification

- Chaque notification doit répondre lorsqu'elle est exploitée, généralement en lançant une activité dans votre application.
- Définissez une intention de contenu à l'aide de la méthode `setContentIntent ()`.
- Passez l'intention emballée dans un objet `PendingIntent`.

# Notification action buttons



- Les boutons d'action peuvent effectuer diverses actions pour le compte de votre application, telles que démarrer une tâche en arrière-plan, passer un appel téléphonique, etc.
- À partir d'Android 7.0 (API niveau 24), répondez aux messages directement à partir des notifications.
- Pour ajouter un bouton d'action, transmettez un PendingIntent à la méthode **addAction ()**.

# Pending intents

- Un PendingIntent est une description d'une intention et d'une action cible à exécuter avec.
- Attribuez un objet PendingIntent à une autre application pour lui donner le droit d'effectuer l'opération que vous avez spécifiée comme si l'autre application était vous-même.

# Méthodes pour créer un PendingIntent

- Pour instancier un PendingIntent, appliquez l'une des méthodes suivantes:
  - PendingIntent.getActivity ()
  - PendingIntent.getBroadcast ()
  - PendingIntent.getService ()

# Arguments de la méthode PendingIntent

1. Contexte d'application
2. Code de demande - ID entier constant pour l'intention en attente
3. Intention d'être délivré
4. L'indicateur PendingIntent détermine comment le système gère plusieurs intentions en attente provenant de la même application.

# Etape 1: Créer un intent

```
Intent notificationIntent =  
    new Intent(this, MainActivity.class);
```

## Etape 2: Créer une instance de PendingIntent

```
Intent notificationIntent =  
    new Intent(this, ListeActivity.class);
```

```
PendingIntent notificationPendingIntent =  
    PendingIntent.getActivity(  
        this, NOTIFICATION_ID,  
        notificationIntent,  
        PendingIntent.FLAG_UPDATE_CURRENT);
```

# Etape 3: Ajouter au notification builder

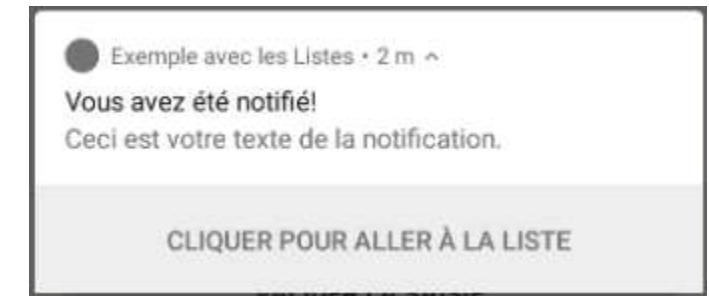
Pour définir l'action tap sur la notification:

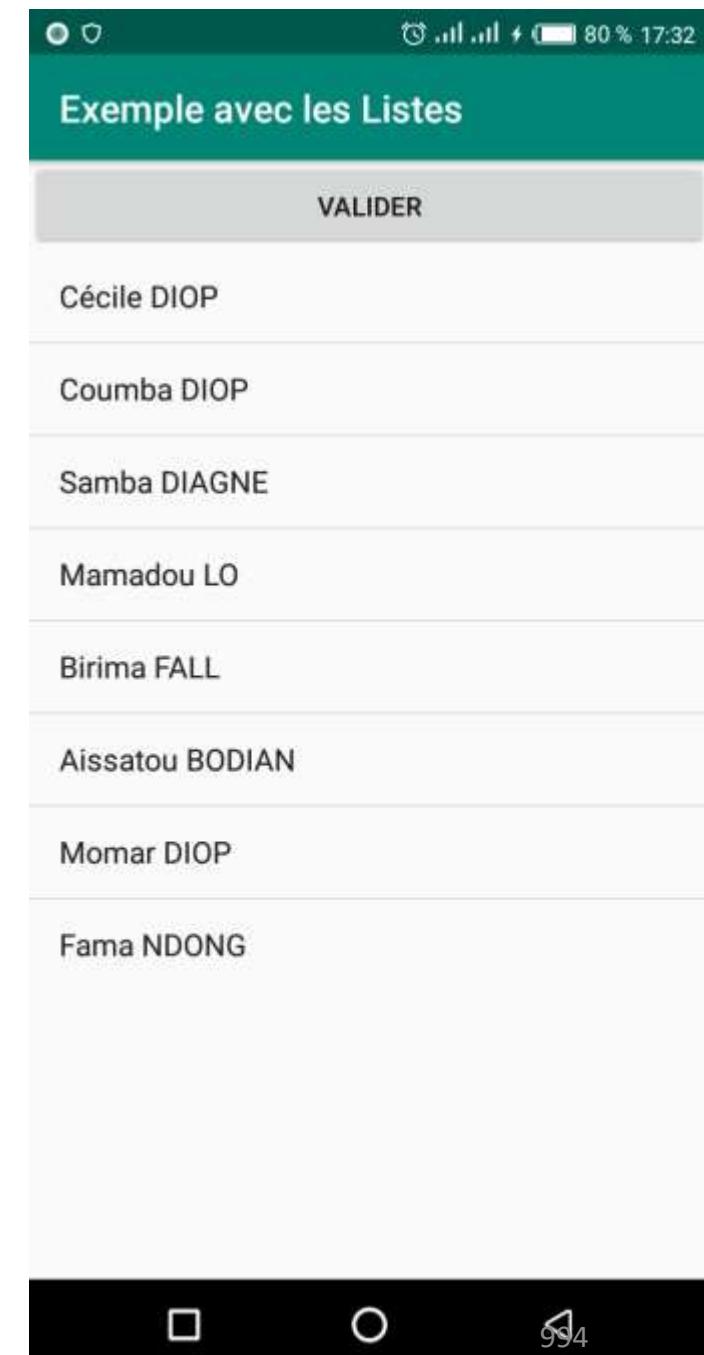
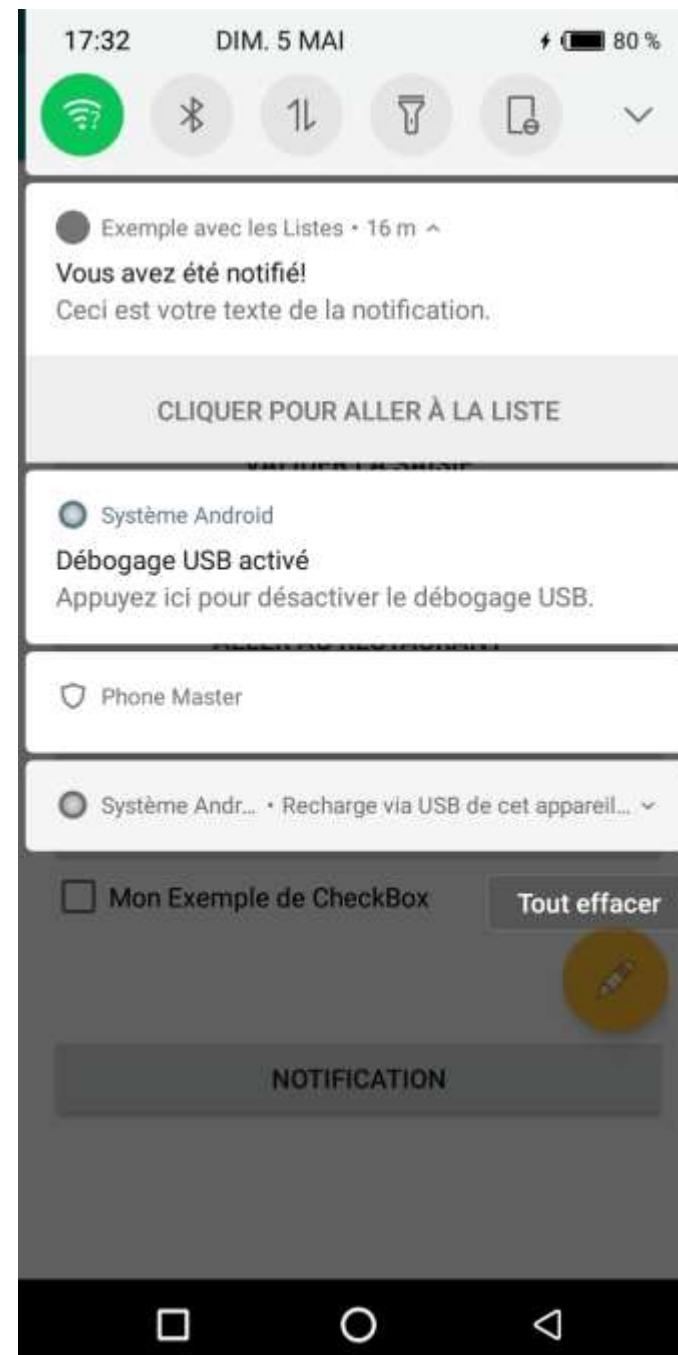
```
NotificationCompat.Builder mBuilder =new NotificationCompat.Builder( context: this, CHANNEL_ID)
    .setSmallIcon(R.mipmap.ic_launcher_round)
    .setAutoCancel(true)
    .setContentTitle("Vous avez été notifié!")
    .setContentText("Ceci est votre texte de la notification.")
    .setContentIntent(notificationPendingIntent);
```

# Ajouter les boutons d'actions

- Utiliser NotificationCompat.Builder.addAction()
  - arguments: icon, caption, PendingIntent

```
NotificationCompat.Builder mBuilder =new NotificationCompat.Builder( context: this, CHANNEL_ID)
    .setSmallIcon(R.mipmap.ic_launcher_round)
    .setAutoCancel(true)
    .setContentTitle("Vous avez été notifié!")
    .setContentText("Ceci est votre texte de la notification.")
    .setContentIntent(notificationPendingIntent)
    .addAction(R.drawable.baseline_schedule_black_18dp,
        title: "Cliquer pour aller à la liste", notificationPendingIntent);
```

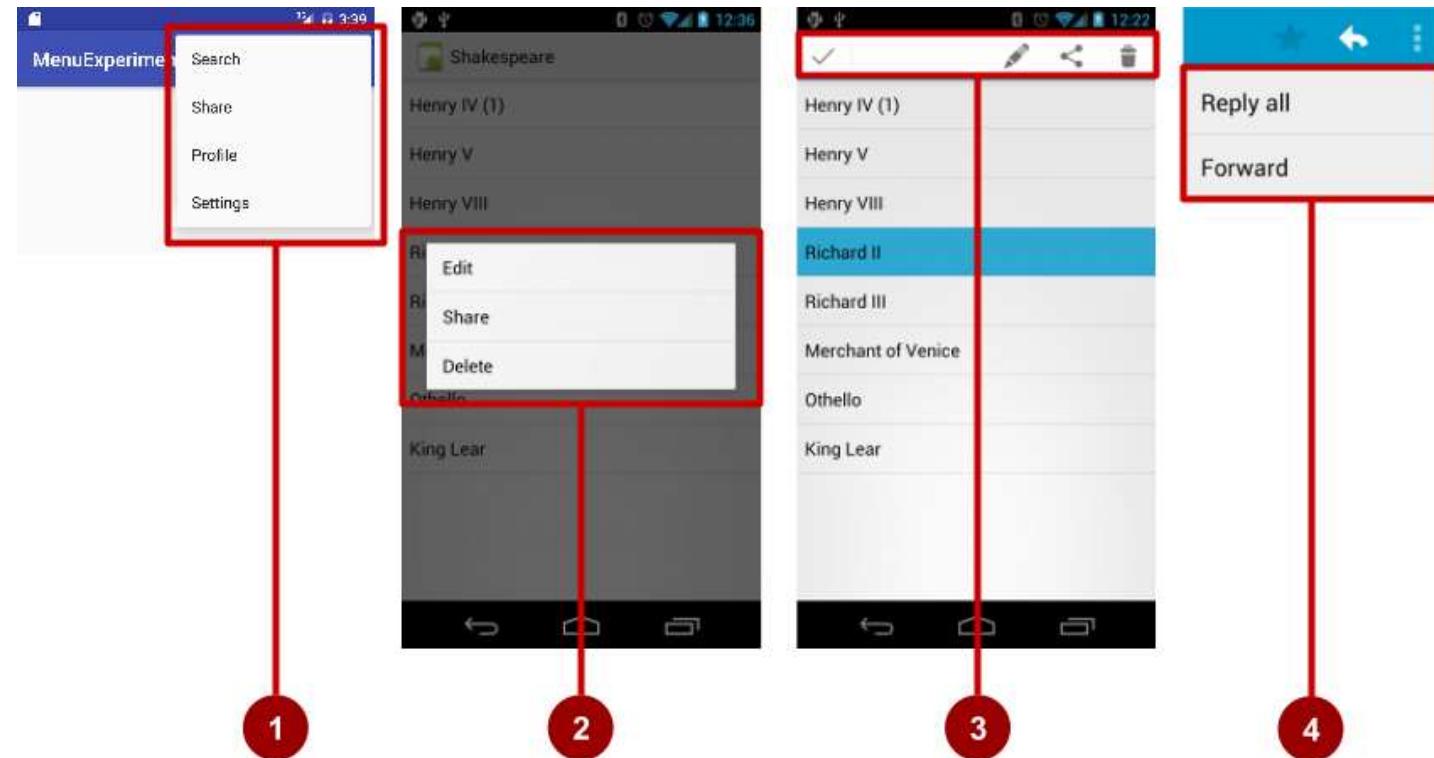




# LES MENUS

# Menus

- Barre d'Action avec des options de menu
- menu Contextuel
- Action bar Contextuelle
- Popup menu

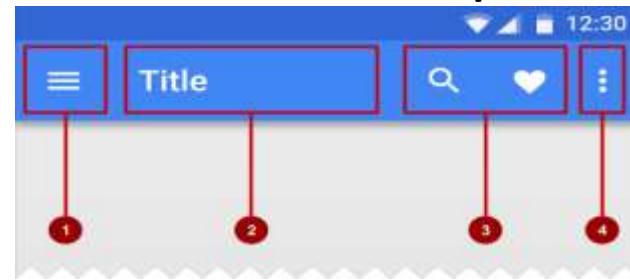


# Barre de menu avec options:

## Qu'est-ce qu'une barre de menus ?

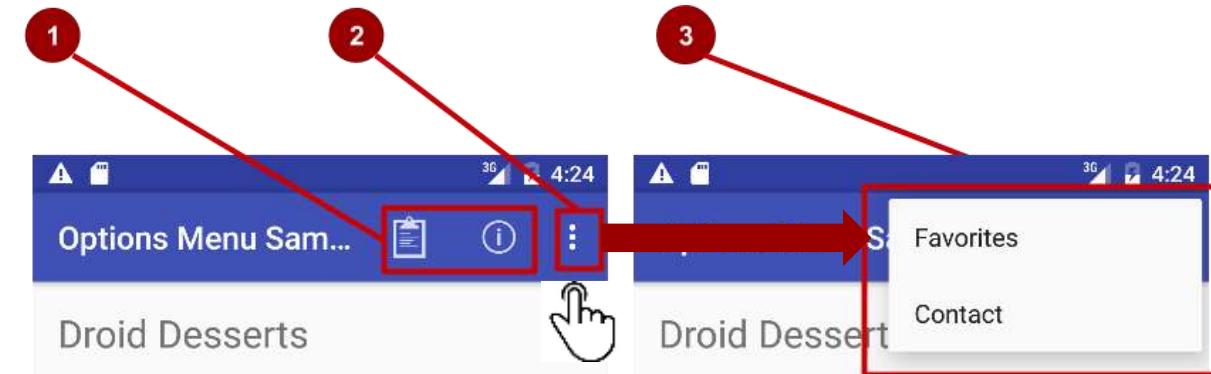
Barre au dessous des écrans — identique quelque soit le téléphone

1. L'icône de navigation permet d'ouvrir le navigation drawer
2. Titre de l'activité courante
3. Icons des items des options du menu
4. Boutons d'actions pour le reste des options menu



# Qu'est-ce que les options du menu

- Icônes d'action dans la barre d'applications pour les éléments importants (1)
- Appuyez sur les trois points, sur le "bouton de débordement d'action" pour afficher le menu d'options (2).



- Apparaît dans le coin droit de la barre d'applications (3)
- Pour naviguer vers d'autres activités et modifier les paramètres de l'application

# Ajouter les options du Menu: Étapes pour implémenter le menu d'options

1. Ressource de menu XML (menu\_main.xml)
2. onCreateOptionsMenu () pour afficher le menu
3. Attribut onClick ou onOptionsItemSelected ()
4. Méthode pour gérer le clic sur un élément

# Créer le menu

1. Créer un répertoire de ressources de menu
2. Créer une ressource de menu XML (menu\_main.xml)
3. Ajouter une entrée pour chaque élément de menu (**Paramètres et Favoris**):

```
<item android:id="@+id/option_settings"
      android:title="Settings" />

<item android:id="@+id/option_favorites"
      android:title="Favorites" />
```

# Inflate options menu

Surcharger onCreateOptionsMenu() dans l'Activité

```
@Override  
public boolean onCreateOptionsMenu(Menu menu) {  
    getMenuInflater().inflate(R.menu.menu_main, menu);  
    return true;  
}
```

# Ajouter les attributs des items du Menu

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/option_settings"
        android:icon="@android:drawable/ic_menu_manage"
        android:orderInCategory="30"
        app:showAsAction="ifRoom"
        android:title="Paraméters" />
    <item android:id="@+id/optionFavorites"
        android:title="Favoris"
        android:icon="@android:drawable/btn_star_big_on"
        android:orderInCategory="30"
        app:showAsAction="ifRoom"
        />
</menu>
```



# Ajouter les attributs des items du Menu

- L'attribut `showAsAction` permet de spécifier le comportement de l'élément dans une `ActionBar`, il peut posséder les valeurs suivantes :
  - `ifRoom` : L'élément sera ajouté aux actions principales de l'`ActionBar` si une place est disponible
  - `never` : Ne jamais rajouter l'action aux actions principales de l'`ActionBar`
  - `always` : Toujours rajouter l'action aux actions principales de l'`ActionBar`. Cette valeur n'est pas conseillé car peut entraîner une superposition d'élément (si nombre de place disponible < nombre d'élément ajouté à l'`ActionBar`), préférez la valeur `ifRoom`.
  - `withText` : Toujours afficher le texte représentant l'action

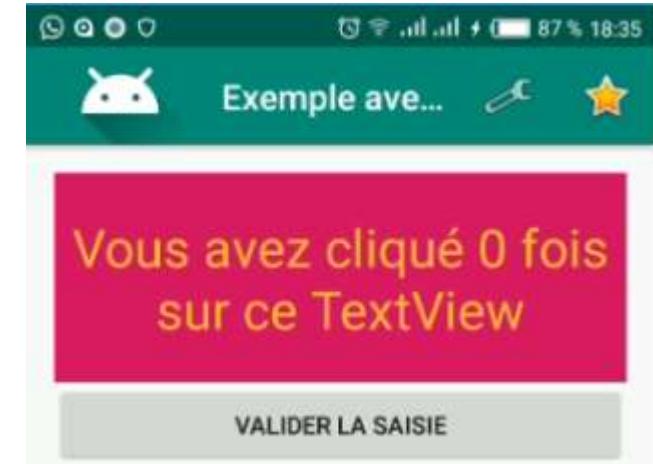
# Override onOptionsItemSelected()

```
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
    switch (item.getItemId()) {  
        case R.id.option_settings:  
            showSettings();  
            return true;  
        case R.id. optionFavorites:  
            showFavorites();  
            return true;  
        default:  
            return super.onOptionsItemSelected(item);  
    }  
}
```

# Navigation via l'icone de votre application

```
ActionBar actionBar = getSupportActionBar();
actionBar.setDisplayHomeAsUpEnabled(true);
actionBar.setHomeAsUpIndicator(R.drawable.ic_launcher_foreground);

public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home:
            if(drw.isDrawerOpen(Gravity.START)==false)
                drw.openDrawer(Gravity.START);
            else
                drw.closeDrawer(Gravity.START);
            return true;
    }
}
```

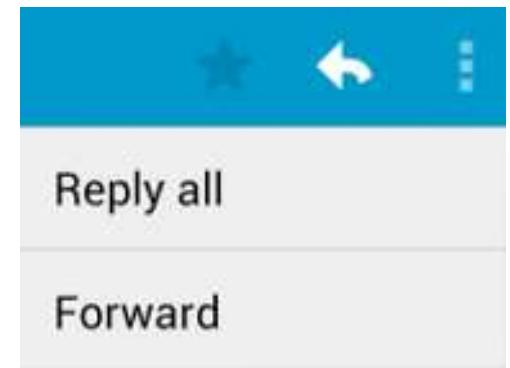


# Autres

- Menus Contextuels
- Barre de menu Contextuelle
- Menu Popup

# POPUP MENU

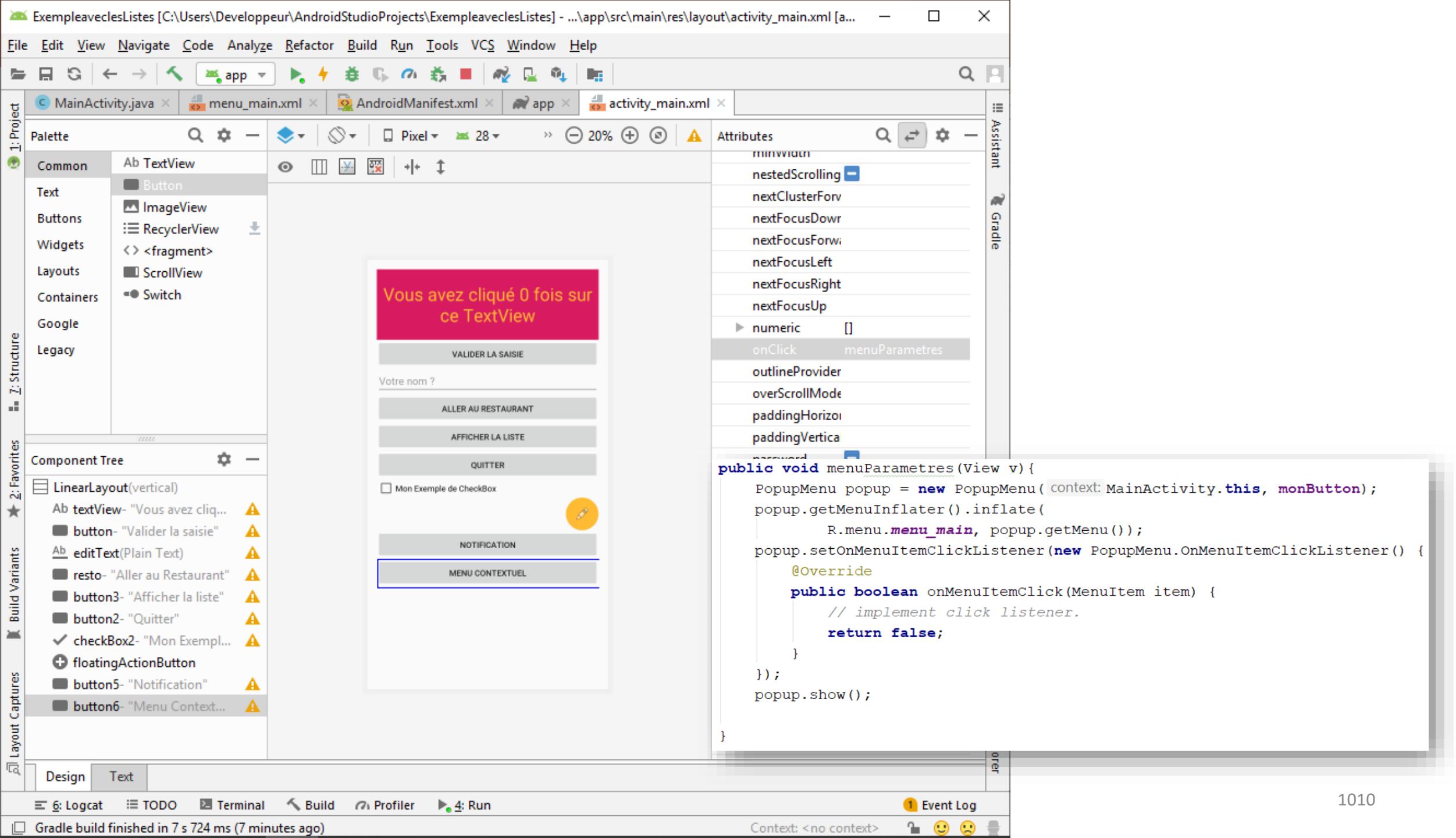
# Qu'est-ce qu'un popup menu?



- Liste verticale d'éléments ancrés à une vue
- Généralement ancré à une icône visible
- Les actions ne doivent pas affecter directement le contenu de la vue
  - Icône du menu Options qui ouvre le menu Options
  - Dans l'application de messagerie, Répondre à tous et Transférer font référence à un message électronique, mais n'affectent ni n'agissent sur le message.

# 1. Etapes

1. Créer un fichier de ressources de menu XML et attribuer des attributs d'apparence et de position
2. Ajouter ImageButton pour l'icône de menu contextuel dans le fichier de présentation d'activité XML
3. Assigner onClickListener à ImageButton
4. Remplacez onClick () pour gonfler la popup et l'enregistrer avec onMenuItemClickListener ()
5. Implémenter onMenuItemClick ()
6. Créer une méthode pour effectuer une action pour chaque élément de menu contextuel



# BOUTONS DE SÉLECTION DE DATES ET HEURES

# Choix de date et d'heure

- DatePicker
  - android:startYear Pour définir l'année de départ du calendrier
- affiché
  - android:endYear Pour définir l'année de fin du calendrier affiché
  - android:minDate Pour définir la date affichée de départ du calendrier sous la forme mm/jj/aaaa
  - android:maxDate Pour définir la date affichée de fin du calendrier sous la forme mm/jj/aaaa
- TimePicker

# Choix de date et d'heure: exemple

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <TextView android:id="@+id/dateAndTime"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        />
    <Button android:id="@+id/dateBtn"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text= " Régler la date"
        />
    <Button android:id="@+id/timeBtn"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text= "Régler l'heure"
        />
</LinearLayout>
```



# Choix de date et d'heure: exemple

```
public class ChronoDemo extends Activity {
    DateFormat fmtDateAndTime=DateFormat.getDateInstance();
    TextView dateAndTimeLabel;
    Calendar dateAndTime=Calendar.getInstance();
    DatePickerDialog.OnDateSetListener d=new DatePickerDialog.OnDateSetListener() {
        public void onDateSet(DatePicker view, int year, int monthOfYear,
            int dayOfMonth) {
            dateAndTime.set(Calendar.YEAR, year);
            dateAndTime.set(Calendar.MONTH, monthOfYear);
            dateAndTime.set(Calendar.DAY_OF_MONTH, dayOfMonth);
            updateLabel();
        }
    };
    TimePickerDialog.OnTimeSetListener t=new TimePickerDialog.OnTimeSetListener(){
        public void onTimeSet(TimePicker view, int hourOfDay,
            int minute) {
            dateAndTime.set(Calendar.HOUR_OF_DAY, hourOfDay);
            dateAndTime.set(Calendar.MINUTE, minute);
            updateLabel();
        }
    };
}
```

# Choix de date et d'heure: exemple

```
@Override  
public void onCreate(Bundle icicle) { super.onCreate(icicle);  
setContentView(R.layout.main);  
  
Button btn=(Button)findViewById(R.id.dateBtn);  
  
btn.setOnClickListener(new View.OnClickListener() { public void onClick(View v) {  
new DatePickerDialog(ChronoDemo.this, d,  
dateAndTime.get(Calendar.YEAR), dateAndTime.get(Calendar.MONTH),  
dateAndTime.get(Calendar.DAY_OF_MONTH)).show();  
}  
});  
  
btn=(Button)findViewById(R.id.timeBtn);  
  
btn.setOnClickListener(new View.OnClickListener() { public void onClick(View v) {  
new TimePickerDialog(ChronoDemo.this, t,  
dateAndTime.get(Calendar.HOUR_OF_DAY), dateAndTime.get(Calendar.MINUTE), true).show();  
}  
});  
  
dateAndTimeLabel=(TextView)findViewById(R.id.dateAndTime); updateLabel();  
}  
  
private void updateLabel() { dateAndTimeLabel.setText(fmtDateAndTime  
.format(dateAndTime.getTime()));  
}
```

# ProgressBar

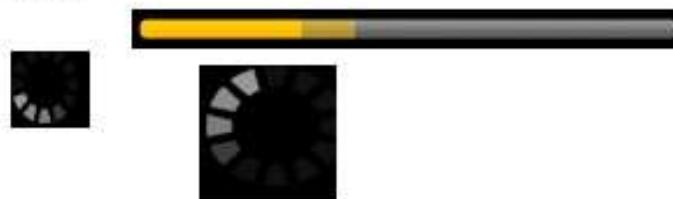
- Deux comportements selon que l'on connaît ou pas la valeur maximale:  
`android:indeterminate` Pour définir le type de progressBar (true=indéterminé, false=déterminé).
- Animation (si indéterminé): `android:indeterminateBehavior="i"` (où i peut être : `repeat` ou `cycle`) définit le comportement de l'animation pour le type circulaire (repeat=recommence l'animation, cycle=changer le sens de l'animation)
- Dimensions
  - `android:maxHeight="unité"`
  - `android:minHeight="unité"`
  - `android:maxWidth="unité"`
  - `android:minWidth="unité"`
- Valeurs (si déterminé)
  - `android:max` Pour définir la valeur maximale
  - `android:progress` Pour définir la valeur initiale
  - `android:secondaryProgress` Pour définir une valeur secondaire (par exemple celle d'un buffer comme on le voit sur des vidéos en streaming)

# Formes des ProgressBar

- En l'absence de paramètre `style` la forme est circulaire
- Pour obtenir d'autres formes on utilise le paramètre `style` :

- `style="?android:attr/s"` où s peut être :

- `progressBarStyleHorizontal`
    - `progressBarStyleSmall`
    - `progressBarStyleLarge`



- On ne peut pas changer la couleur

## SeekBar



C'est un ProgressBar sous forme de barre horizontale dotée d'un curseur permettant de modifier la valeur si on a choisi `android:indeterminate="false"` sinon le curseur ne marche pas et la barre bouge sans arrêt.

# Exemple ProgressBar

```
<ProgressBar android:id="@+id/progres"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:minWidth="300px"
    android:minHeight="30px"
    android:max="100"
    android:progress="30"
    android:secondaryProgress="40"
    android:indeterminate="false"
    style="?android:attr/progressBarStyleHorizontal"/>
```

# RatingBar



Paramètres :

`android:isIndicator` Pour indiquer si l'utilisateur peut modifier la valeur ou pas (true= non modifiable)

`android:numStars` Pour définir le nombre d'étoiles affichées

`android:rating` Pour définir la position initiale

`android:stepSize` Pour définir le pas de progression (on peut colorier des  $\frac{1}{4}$  d'étoiles par exemple)

# Horloges et Chronomètres

AnalogClock



DigitalClock

4:58:17 pm

Chronometer

Ce cours a commencé depuis 51:45 déjà

[android:format](#)="f" (où f est une chaîne dans laquelle la première occurrence de %s sera remplacée par la valeur du chronomètre sous la forme MM:SS ou H:MM:SS)

# AnalogClock

```
<AnalogClock android:id="@+id/horloge"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
/>
```

# Sélection Date/heure: Date

- Android supporte également les widgets (**DatePicker**, **TimePicker**) et boîtes de dialogue (**DatePickerDialog**, **TimePickerDialog**) pour aider les utilisateurs entrer des dates et des heures.
- Le **DatePicker** et **DatePickerDialog** permettent de régler la date de début de sélection, sous la forme d'une année, mois et jour. Valeur du mois traverse de 0 pour janvier 11 pour le 2 décembre décembre.
- Chaque widget fournit un objet de rappel **OnDateChangedListener** ou **OnDateSetListener**) où vous êtes informé d'une nouvelle date sélectionnée par l'utilisateur.

# Sélection Date/heure: Heure

**TimePicker** et **TimePickerDialog** permettent de:

1. régler le temps initial, l'utilisateur peut ajuster, sous la forme d'une **heure** (0 à 23) et une **minute** (0 à 59)
2. *indiquer si la sélection doit être en mode 12-heure* (AM/PM), ou **mode 24-heure**.
3. fournir un objet de rappel (**OnTimeChangedListener** or **OnTimeSetListener**) pour être averti des lorsque l'utilisateur a choisi une nouvelle fois (ce qui vous est fournie sous la forme d'une heure et minute)

# Sélection Date/heure



# Sélection Date/heure: exemple

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:id="@+id/widget28"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
>
    <TextView
        android:id="@+id/lblDateAndTime"
        android:layout_width="fill_parent"
        android:layout_height="47px"
        android:background="#ff000099"
        android:textStyle="bold"
    >
    </TextView>
    <Button
        android:id="@+id/btnDate"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Set the Date"
    >
    </Button>
    <Button
        android:id="@+id/btnTime"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Set the Time"
    >
</LinearLayout>
```



# Sélection Date/heure: exemple

```
package cis493.demoui;
import android.app.Activity;
import android.os.Bundle;
import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.TimePicker;
import android.widget.TextView;
import java.text.DateFormat;
import java.util.Calendar;
```

```
public class AndDemoUI extends Activity {
    DateFormat fmtDateAndTime = DateFormat.getDateInstance();
    TextView lblDateAndTime;
    Calendar myCalendar = Calendar.getInstance();
```



```
DatePickerDialog.OnDateSetListener d = new DatePickerDialog.OnDateSetListener()
{
    public void onDateSet(DatePicker view,
                          int year, int monthOfYear, int dayOfMonth) {
        myCalendar.set(Calendar.YEAR, year);
        myCalendar.set(Calendar.MONTH, monthOfYear);
        myCalendar.set(Calendar.DAY_OF_MONTH, dayOfMonth);
        updateLabel();
    }
};
```

# Sélection Date/heure: exemple

```
TimePickerDialog.OnTimeSetListener t = new TimePickerDialog.OnTimeSetListener()
{
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        myCalendar.set(Calendar.HOUR_OF_DAY, hourOfDay);
        myCalendar.set(Calendar.MINUTE, minute);
        updateLabel();
    }
};

private void updateLabel() {
    lblDateAndTime.setText(fmtDateAndTime.format(myCalendar.getTime()));
}
```



# Sélection Date/heure: exemple

```
@Override
public void onCreate(Bundle icicle) {
    super.onCreate(icicle);
    setContentView(R.layout.main);
    lblDateAndTime = (TextView) findViewById(R.id.lblDateAndTime);
    Button btnDate = (Button) findViewById(R.id.btnDate);
    btnDate.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            new DatePickerDialog(AndDemoUI.this, d,
                myCalendar.get(Calendar.YEAR),
                myCalendar.get(Calendar.MONTH),
                myCalendar.get(Calendar.DAY_OF_MONTH)).show();
        }
    });

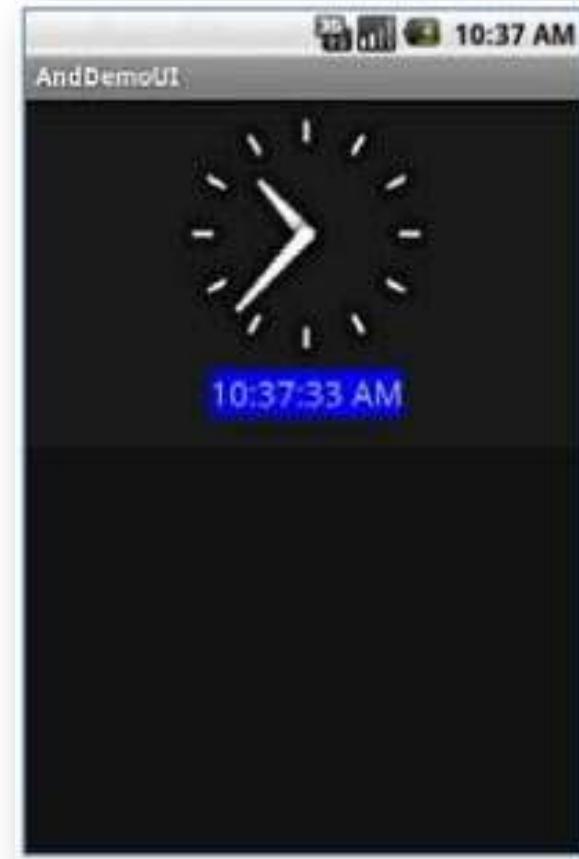
    Button btnTime = (Button) findViewById(R.id.btnTime);
    btnTime.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            new TimePickerDialog(AndDemoUI.this, t,
                myCalendar.get(Calendar.HOUR_OF_DAY),
                myCalendar.get(Calendar.MINUTE), true).show();
        }
    });

    updateLabel();
} // onCreate
} // class
```

# Sélection Date/heure: DigitalClock et AnalogClock

Mettre à jour automatiquement le temps (sans intervention de l'utilisateur est requise).

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    android:id="@+id/widget34"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android"
>
    <DigitalClock
        android:id="@+id/digital"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#ff0000ff"
        android:textSize="20px"
        android:layout_below="@+id/analog"
        android:layout_centerHorizontal="true"
    >
    </DigitalClock>
    <AnalogClock
        android:id="@+id/analog"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
    >
    </AnalogClock>
</RelativeLayout>
```



# Onglets

- Une Activity peut être subdivisée en plusieurs onglets
- Afin d'utiliser des onglets, il vous faut les prochaines pièces:
  - TabHost : le conteneur surplombant les boutons d'onglets et leur contenus
  - TabWidget : la rangée de boutons d'onglets qui contient le texte
    - et optionnellement les icônes
  - FrameLayout : le conteneur pour les contenus des onglets, chaque onglet est un enfant du FrameLayout

# Onglets : quelques règles

Afin que les onglets fonctionnent correctement, il faut suivre certaines règles:

- Le TabWidget doit avoir une **android:id qui est @android:id/tabs**
- Il est conseillé de mettre du padding entre le FrameLayout et les tab buttons, afin que le contenu ne «colle» pas aux tab buttons.
- Il est possible d'utiliser une TabActivity, mais il faut alors donner au TabHost une **android:id of @android:id/tabhost**
- TabActivity regroupe de la fonctionnalité propre aux onglets et facilite l'usage d'onglets

# Exemple de layout avec onglets

```
<TabHost android:id="@+id/tabhost"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
        <TabWidget android:id="@+id/tabs"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
        />
        <FrameLayout android:id="@+id/tabcontent"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent">
            <AnalogClock android:id="@+id/tab1"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:layout_centerHorizontal="true"
            />
            <Button android:id="@+id/tab2"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text="A semi-random button"
            />
        </FrameLayout>
    </LinearLayout>
</TabHost>
```

# Onglets : comment faire

- Il est impératif d'utiliser du code Java afin de dire au **TabHost quelle view** représente le contenu d'un bouton mais aussi à quoi ressemblera un bouton
- Pour ceci, nous utilisons un object **TabSpec**: en utilisant la fonction **newTabSpec() sur le TabHost**
- Il y a ensuite deux fonctions clés:
  - **setContent()** pour indiquer le contenu d'un onglet
  - android:id de la view
  - ou un Intent (contenu d'une autre Activity, par exemple)
- **setIndicator()**
  - texte pour l'onglet
  - image drawable pour l'icone
  - view
- When tabs are ready, call the `setup()` method on the tabhost and add the tabs

# Onglets : exemple d'activity

```
public class TabDemo extends Activity{  
    @Override  
    public void onCreate(Bundle icicle) {  
        super.onCreate(icicle);  
        setContentView(R.layout.main);  
        TabHost tabs=(TabHost)findViewById(R.id.tabhost);  
  
        tabs.setup();  
  
        TabHost.TabSpec spec=tabs.newTabSpec("tag1");  
  
        spec.setContent(R.id.tab1);  
        spec.setIndicator("Clock");  
        tabs.addTab(spec);  
  
        spec=tabs.newTabSpec("tag2");  
        spec.setContent(R.id.tab2);  
        spec.setIndicator("Button");  
        tabs.addTab(spec);  
    }  
}
```



# Exercice 3

- Créez une application avec deux onglets
  - Le premier onglet reprend le layout créé précédemment avec «pierre, papier, ciseaux» créez précédemment
  - Le deuxième onglet contient une TextView qui affiche «ceci est onglet 2»

# Développement d'Applications Natives avec Android

## GÉRER LA CONNEXION INTERNET



# Gérer la connexion réseau internet



# Required Permissions

```
<uses-permission android:name="android.permission.INTERNET" />
```

```
<uses-permission  
    android:name="android.permission.ACCESS_NETWORK_STATE" />
```

# Getting Network information

- ConnectivityManager
  - Répond à des questions sur l'état de la connectivité réseau
  - Notifie les applications lorsque la connectivité réseau change
- NetworkInfo
  - Décrit le statut d'une interface réseau d'un type donné
  - Mobile ou Wi-Fi

# Vérifier si le réseau est disponible

```
ConnectivityManager connMgr =  
(ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);  
NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();  
String str="Aucune connexion disponible !!!";  
if (networkInfo != null && networkInfo.isConnected()) {  
    str="Vous êtes connecté par " + networkInfo.getTypeName()  
        +". Les infos sont"+networkInfo.getExtraInfo();  
}  
Toast.makeText(getApplicationContext(),str,Toast.LENGTH_LONG).show();
```

# Vérifier le WiFi et le mobile

```
NetworkInfo networkInfo =  
    connMgr.getNetworkInfo(ConnectivityManager.TYPE_WIFI);  
boolean isWifiConn = networkInfo.isConnected();  
  
networkInfo =  
    connMgr.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);  
boolean isMobileConn = networkInfo.isConnected();
```

# Utiliser les Thread

- AsyncTask—tâche très courte ou aucun résultat renvoyé à l'interface utilisateur
- AsyncTaskLoader—pour des tâches plus longues, renvoie le résultat à l'interface utilisateur
- Background Service
- N'implémentez pas de codes liés au réseau dans le thread d'interface utilisateur!

# AsyncTask

- Seul un thread d'interface utilisateur peut créer et démarrer une AsyncTask.
- execute (): démarre AsyncTasks en contexte série
- Concurrent avec le thread d'interface utilisateur, mais séquentiel avec d'autres threads de travail.
- executeOnExecutor (AsyncTask.THREAD\_POOL\_EXECUTOR)
- En même temps que l'interface utilisateur et les threads de travail.

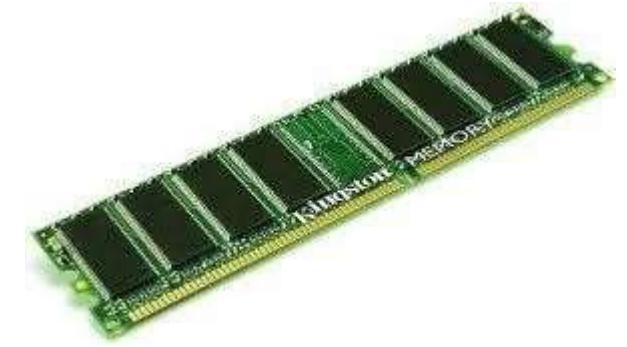
# Développement d'Applications Natives avec Android

**ANDROID- GÉRER LA PERSISTANCE DES DONNÉES:  
FICHIERS ET BASES DE DONNÉES**



# LES FICHIERS

# Les Fichiers sous Android



- Android utilise les mêmes systèmes de constructions de fichiers que dans une application java typique.
- Les fichiers peuvent être stockés dans tout de type de support de stockage(carte SD,...). Ils peuvent également être obtenus à partir du réseau (comme nous le verrons plus tard).
- Les fichiers stockés dans la mémoire de l'appareil, sont ensemble avec les autres ressources de l'application (tels que les icônes, images, musique, ...).
- Nous appellerons ce type: fichiers de ressources.



# Les Fichiers sous Android

- Les options de stockage de données sont
- les suivantes:
  1. Shared Preferences pour sauvegarder les données sous la forme de paires clé-valeur.
  2. Support de stockage interne
  3. Support de stockage externe
  4. SQLiteDatabases
  5. En utilisant le réseaux pour une sauvegarde sur le web par exemple

# Les Fichiers sous Android

## Accéder au SD card

```
String spPath = Environment.getExternalStorageDirectory()  
    .getAbsolutePath() + "/myFileName.txt";
```



# Les Fichiers sous Android

```
InputStream is = this.getResources()  
    .openRawResource(R.drawable.mon_fichier);
```

The screenshot shows the Android Studio Package Explorer with a project named "TestFichiers". The "res" directory is expanded, showing subfolders for different screen densities: "drawable-hdpi", "drawable-ldpi", "drawable-mdpi", "drawable-xhdpi", "layout", "menu", "values", "values-v11", and "values-v14". It also contains the "AndroidManifest.xml" file, icons for the launcher ("ic\_launcher.png" and "ic\_launcher-web.png"), and configuration files like "proguard-project.txt" and "project.properties". An arrow points from the "mon\_fichier.txt" file located in the "drawable-hdpi" folder to a second window titled "mon\_fichier.txt". This second window displays the following text:

```
1 Best Poster Award at ESWC 2012!  
2  
3 Reblogged from VIDEBO: The ESWC 2012 conference  
ended with a bang for me and the rest of  
4 the W4RA team: We were awarded the Best Poster Award  
for our poster "Bringing the Web of  
5 Data to Developing Countries: Linked Market Data in  
the Sahel". You can find the poster abstract  
6 here and the poster itself here.
```

# Les Fichiers sous Android



The image shows a screenshot of an Android application running on an emulator. The title bar says "TestFichiers". A toast message is displayed in the center of the screen with the text:

Best Poster Award at ESWC 2012!  
Reblogged from VIDEBO: The ESWC 2012 conference ended with a bang for me and the rest of the W4RA team: We were awarded the Best Poster Award for our poster "Bringing the Web of Data to Developing Countries: Linked Market Data in the Sahel". You can find the poster abstract here and the poster itself here.

On the left, there is a code editor window titled "MainTestFichiersActivity.java" showing Java code for an Android application. The code imports various Java and Android classes and defines a MainTestFichiersActivity class that overrides the onCreate method to set the content view and handle file operations.

```
package com.example.testfichiers;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.widget.Toast;

public class MainTestFichiersActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_test_fichiers);
        try {
            PlayWithRawFiles();
        } catch (IOException e) {
            Toast.makeText(getApplicationContext(),"Problèmes: " + e.getMessage(), 1).show();
        }
    }
}
```

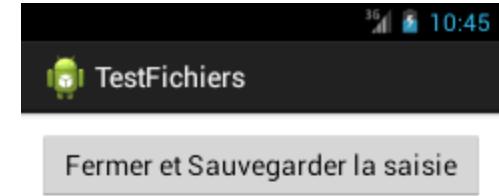
# Les Fichiers sous Android

```
public void PlayWithRawFiles() throws IOException {
    String str = "";
    StringBuffer buf = new StringBuffer();
    InputStream is = this.getResources().openRawResource(R.drawable.mon_fichier);
    BufferedReader reader = new BufferedReader(new InputStreamReader(is));
    if (is != null) {
        while ((str = reader.readLine()) != null) {
            buf.append(str + "\n");
        }
    }
    is.close();
    Toast.makeText(getApplicationContext(), buf.toString(), Toast.LENGTH_LONG).show();
}// PlayWithRawFiles
```

# Les Fichiers sous Android

## Exemple n°2

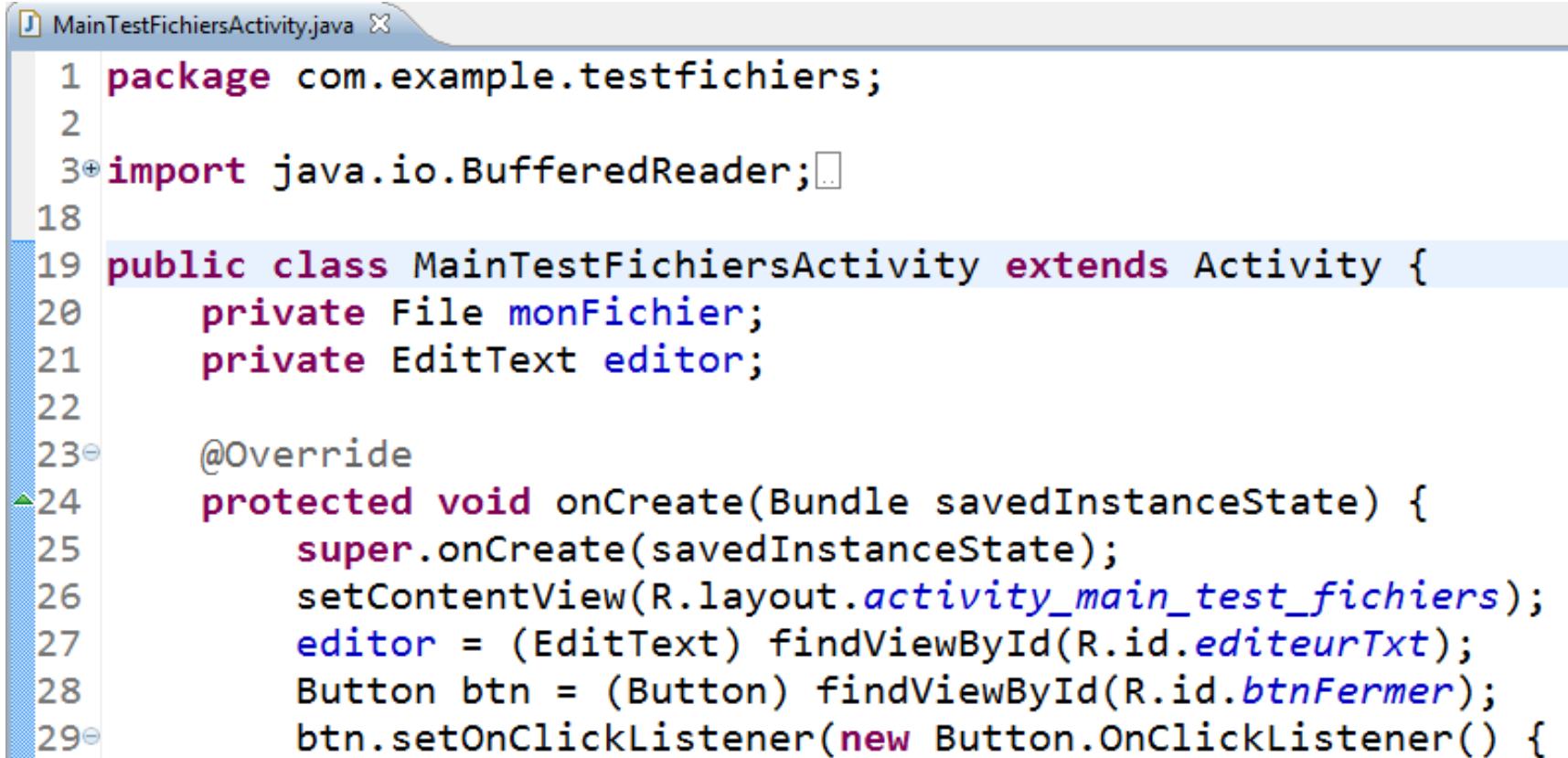
### Sauvegarde en interne



Last Resort est une série télévisée américaine en treize épisodes de 42 minutes créée par Shawn Ryan et diffusée entre le 27 septembre 2012 et le 24 janvier 2013 sur le réseau ABC2 aux États-Unis et en simultané sur le réseau Global3 au Canada.

# Les Fichiers sous Android

## Exemple n°2



```
J MainTestFichiersActivity.java X
1 package com.example.testfichiers;
2
3 import java.io.BufferedReader;...
18
19 public class MainTestFichiersActivity extends Activity {
20     private File monFichier;
21     private EditText editor;
22
23@Override
24 protected void onCreate(Bundle savedInstanceState) {
25     super.onCreate(savedInstanceState);
26     setContentView(R.layout.activity_main_test_fichiers);
27     editor = (EditText) findViewById(R.id.editeurTxt);
28     Button btn = (Button) findViewById(R.id.btnFermer);
29     btn.setOnClickListener(new Button.OnClickListener() {
```

# Les Fichiers sous Android

## Exemple n°2

```
public void onClick(View v) {
    try {// on déclare notre futur fichier
        monFichier = new File(Environment.getExternalStorageDirectory()
            + File.separator+ "appli_test", "testAppli.txt");
        // pour créer le répertoire dans lequel mettre le fichier
        File monRep = new File(Environment.getExternalStorageDirectory()
            + File.separator+ "appli_test");
        Boolean success = true;
        if (!monRep.exists()){// On crée le répertoire (s'il n'existe pas!!)
            success = monRep.mkdir();
        }
        if (success) {
            String data = editor.getText().toString();
            // true pour écrire en fin de fichier, et non l'écraser
            FileOutputStream output = new FileOutputStream(monFichier,true);
            output.write(data.getBytes());
        } else {
            Toast.makeText(getApplicationContext(),"ERROR CREA. REP",Toast.LENGTH_LONG).show();
        }
    } catch (Exception e) {
        Toast.makeText(getApplicationContext(), e.getMessage(),Toast.LENGTH_LONG).show();
    }
});
```

# Les Fichiers sous Android

## Exemple n°2

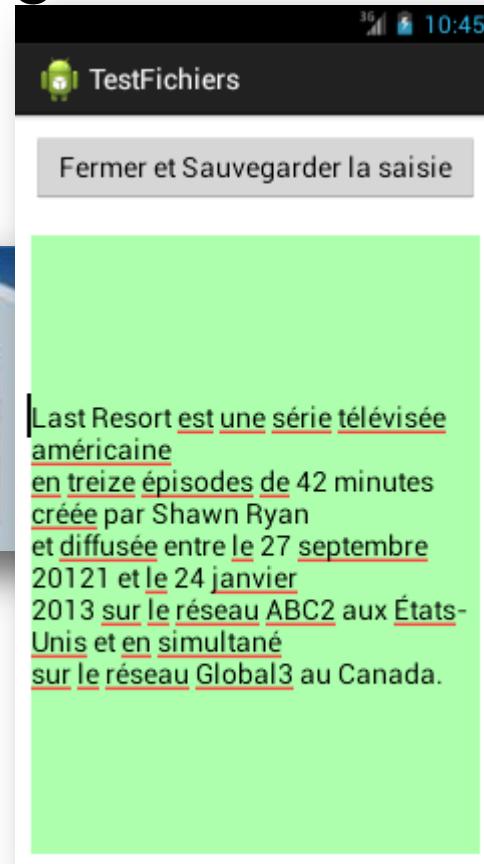
The screenshot shows the DDMS (Dalvik Debug Monitor Service) interface, specifically the File Explorer tab. The window title is "File Explorer". The toolbar at the top includes icons for Threads, Heap, Allocation Tracker, Network Statistics, Emulator Control, Java, and DDMS. Below the toolbar is a menu bar with tabs: Threads, Heap, Allocation Tracker, Network Statistics, Emulator Control, and File Explorer (which is selected). The main area is a file browser with columns for Name, Size, Date, Time, Permissions, and Info.

Name	Size	Date	Time	Permissions	Info
acct		2013-07-19	22:01	drwxr-xr-x	
cache		2013-07-09	22:08	drwxrwx---	
config		2013-07-19	22:01	dr-x-----	
d		2013-07-19	22:01	lrwxrwxrwx	-> /sys/ker...
data		2013-05-12	22:06	drwxrwx--x	
default.prop	116	1970-01-01	00:00	-rw-r--r--	
dev		2013-07-19	22:01	drwxr-xr-x	
etc		2013-07-19	22:01	lrwxrwxrwx	-> /system...
init	109412	1970-01-01	00:00	-rwxr----	
init.goldfish.rc	2487	1970-01-01	00:00	-rwxr----	
init.rc	18247	1970-01-01	00:00	-rwxr----	
init.trace.rc	1795	1970-01-01	00:00	-rwxr----	
init.usb.rc	3915	1970-01-01	00:00	-rwxr----	
mnt		2013-07-19	22:01	drwxrwxr-x	
asec		2013-07-19	22:01	drwxr-xr-x	
obb		2013-07-19	22:01	drwxr-xr-x	
sdcard		2013-07-19	22:06	d---rwxr-x	
LOST.DIR		2013-07-19	13:48	d---rwxr-x	
appli_test		2013-07-19	22:06	d---rwxr-x	
testAppli.txt	82	2013-07-19	22:06	----rwxr-x	
secure		2013-07-19	22:01	drwx-----	
shell		2013-07-19	22:01	drwx-----	
proc		1970-01-01	00:00	dr-xr-xr-x	
root		2012-09-26	18:04	drwx-----	
sbin		1970-01-01	00:00	drwxr-x---	

# Les Fichiers sous Android

## Exemple n°3

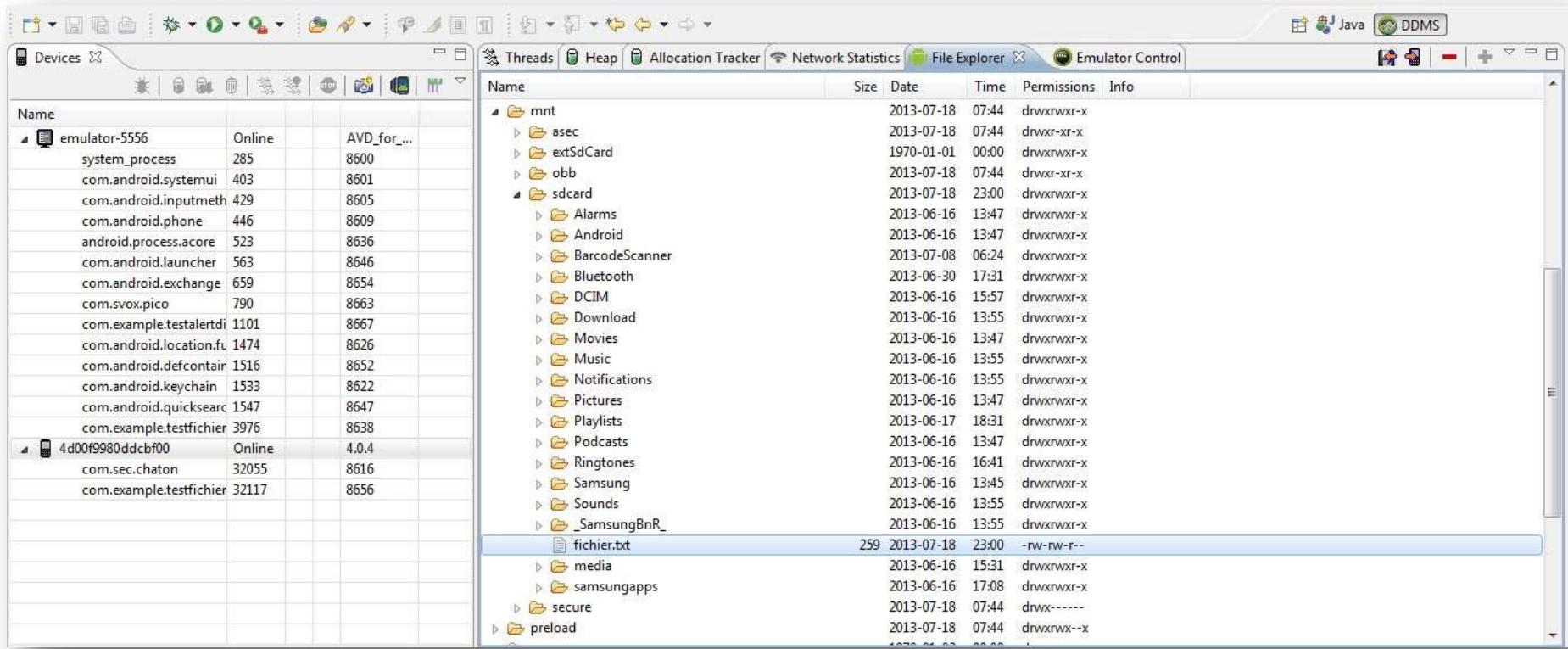
- Sauvegarde dans la carte mémoire



# Les Fichiers sous Android

## Exemple n°3

- Lire/Ecrire dans une mémoire externe



# Les Fichiers sous Android

## Exemple n°3



Since SDK1.6 it is necessary to request permission to write to the SD card.  
Add the following clause to your `AndroidManifest.xml`

```
<uses-permission  
    android:name="android.permission.WRITE_EXTERNAL_STORAGE">  
</uses-permission>
```



# Les Fichiers sous Android

## Exemple n°3

```
private File myFile = new File("/sdcard/fichier.txt");
private EditText editor;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main_test_fichiers);
    editor = (EditText) findViewById(R.id.editeurTxt);
    Button btn = (Button) findViewById(R.id.btnFermer);
    btn.setOnClickListener(new Button.OnClickListener() {
        public void onClick(View v) {
            try {
                myFile.createNewFile();
                FileOutputStream fOut = new FileOutputStream(myFile);
                OutputStreamWriter myOutWriter = new OutputStreamWriter(fOut);
                myOutWriter.append(editor.getText());
                myOutWriter.close();
                fOut.close();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    });
}
```

# BASES DE DONNÉES SQLITE

# Une Base de données Android = une base de données SQLiteDatabase

- Dans le code, une base de données est modélisée par un objet de la classe `android.database.sqlite.SQLiteDatabase`
- Cette classe permet donc d'insérer des données (`insert()`), éditer (`update()`), supprimer (`delete()`), de lancer des requêtes (`avec query()`) ou des requêtes qui ne retournent pas de données par `execSQL()`
- Les requêtes Create, Read, Update, Delete (~ INSERT, SELECT, UPDATE, DELETE de SQL) sont dites des requêtes CRUD

# Une classe Entité

```
public class Etudiant {  
    private int id;  
    private String nom;  
    private String prenom;  
    private String filiere;  
  
    public Etudiant(int id, String nom, String prenom, String filiere) {}  
    public Etudiant(String nom, String prenom, String filiere) {}  
    public int getId() {}  
    public void setId(int id) {}  
    public String getNom() {}  
    public void setNom(String nom) {}  
    public String getPrenom() {}  
    public void setPrenom(String prenom) {}  
    public String getFiliere() {}  
    public void setFiliere(String filiere) {}  
    public String toString() {  
        return "Etudiant n°"+id+" Nom: "+nom+" Prénom: "+prenom+" Filière: "+filiere;  
    }  
}
```

# Une classe d'aide (helper)

- Pour créer et/ou mettre à jour une BD, on écrit une classe qui hérite de la classe abstraite `android.database.sqlite.SQLiteOpenHelper`
- Cela nécessite de créer un constructeur qui appellera un des constructeurs avec argument de `SQLiteOpenHelper` : il n'y a pas de constructeur sans argument dans `SQLiteOpenHelper`
- Le constructeur de `SQLiteOpenHelper` utilisé est `public SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version)`
  - `context` est le contexte de l'application
  - `name` est le nom du fichier contenant la BD
  - `factory` est utilisé pour créer des `Cursor`. En général on met `null`
  - `version` est le numéro de version de la BD (commençant à 1)

# Du helper à SQLiteDatabase

- = de la classe d'aide à la base de données
- Le constructeur précédent est un proxy qui est exécuté rapidement
- La BD sera réellement créée au lancement de `getWritableDatabase()` (pour une base en lecture et écriture) ou de `getReadableDatabase()` (pour une base en lecture seule) sur cet objet de la classe d'aide
- Bref on a :

```
public class EtudiantOpenHelper extends SQLiteOpenHelper {...}  
baseHelper = new EtudiantOpenHelper(context, nomBD, null, 1);  
maBaseDonnees = baseHelper.getWritableDatabase();
```

- et un helper est (évidemment) associé à une base de données. XXX est soit Write soit Read

# Création, mise à jour d'une BD : code du helper

- Sur un objet d'une classe héritant de `SQLiteOpenHelper` (**classe d'aide**), certaines méthodes sont appelées automatiquement :
  - `public void onCreate(SQLiteDatabase db)` est appelée automatiquement par l'environnement d'exécution quand la BD n'existe pas. On met le code de création des tables et leurs contenus dans cette méthode
  - `public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)` quand le numéro de version de la BD a été incrémentée
- Ces deux méthodes sont abstraites dans la classe de base et doivent donc être implémentées dans la classe d'aide
- `maBaseDonnees` de class `SQLiteDatabase` sera obtenu comme retour de `getWritableDatabase()` ou `getReadableDatabase()`

# Code de la classe d'aide EtudiantOpenHelper

```
public class EtudiantOpenHelper extends SQLiteOpenHelper {
    String requeteCreationTable = "Create table Etudiants(id integer primary key autoincrement, " +
        "nom text, prenom text, filiere text);"

    public EtudiantOpenHelper(Context context, String name,
        CursorFactory factory, int version) {
        super(context, name, factory, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(requeteCreationTable);
        Log.i("BD Etudiants", "Table Etudiants crée avec Succés");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // Ici nous supprimons la base et les données pour en créer une nouvelle
        // ensuite. Vous pouvez créer une logique de mise à jour propre à votre base permettant
        // de garder les données à la place.
        db.execSQL("drop table Etudiants;");
        // Création de la nouvelle structure.
        onCreate(db);
    }
}
```

# Insérer des données dans une SQLiteDatabase (1/2)

- Ayant obtenu une SQLiteDatabase, on utilise les méthodes de cette classe pour faire des opérations sur la base de données
- public long insert (String table, String nullColumnHack, ContentValues values) insert dans la table table les valeurs indiquées par values
- values, de classe ContentValues, est une suite de couples (clé, valeur) où la clé, de classe String, est le nom de la colonne et valeur, sa valeur
- Bref on prépare tout, la ligne à insérer en remplissant values par des put() successifs puis on lance insert()
- Exemple :

```
public long insertEtudiant(Etudiant etudiant) {  
    ContentValues valeurs = new ContentValues();  
    valeurs.put("nom", etudiant.getNom());  
    valeurs.put("prenom", etudiant.getPrenom());  
    valeurs.put("filiere", etudiant.getFiliere());  
    Log.i("BD Etudiants", "Etudiant " + etudiant.toString()  
          + " inséré avec Succés");  
    return maBaseDonnees.insert("Etudiants", null, valeurs);  
}
```

# Insérer des données dans une SQLiteDatabase (2/2)

- Le second argument nullColumnHack est le nom de colonne qui aura la valeur NULL si values est vide. Cela est du au fait que SQLite ne permet pas de lignes vides. Ainsi avec cette colonne, au moins un champ dans la ligne aura une valeur (= NULL). Bref cela sert seulement lorsque values est vide !
- Cette méthode insert() retourne le numéro de la ligne insérée ou -1 en cas d'erreur
- En fait cette insertion est le Create (C) de CRUD

# Récupérer des données dans une SQLiteDatabase

- La méthode la plus simple (!) pour récupérer des données (~ SELECT) est : `public Cursor query (String table, String[] columns, String whereClause, String[] selectionArgs, String groupBy, String having, String orderBy)`
  - `columns` est la liste des colonnes à retourner. Mettre null si on veut toutes les colonnes
  - `whereClause` est la clause WHERE d'un SELECT (sans le mot WHERE). Mettre null si on veut toutes les lignes
  - `selectionArgs` est utile si dans `whereClause` (~ WHERE), il y a des paramètres notés ?. Les valeurs de ces paramètres sont indiqués par `selectionArgs`. Bref en général on met null
  - `groupBy` est la clause GROUP BY d'un SELECT (sans les mots GROUP BY). Utile pour des SELECT COUNT(\*). Bref on général on met null
  - `having` indique les groupes de lignes à retourner (comme HAVING de SQL = un WHERE sur résultat d'un calcul, pas sur les données)
  - `orderBy` est la clause ORDER BY d'un SELECT (sans les mots ORDER BY). Mettre null si on ne tient pas compte de l'ordre

# Exemple de requête SELECT pour Android

- Euh, la classe android.database.sqlite.SQLiteQueryBuilder est (semble) faite pour cela
- Voir à <http://sqlite.org/lang.html>
- Sinon, quelques exemples
- Rappel : SELECT peut être obtenu avec : public Cursor query (String table, String[] columns, String whereClause, String[] selectionArgs, String groupBy, String having, String orderBy)
- db.query(TABLE\_CONTACTS, new String[] { KEY\_ID, KEY\_NAME, KEY\_PH\_NO }, KEY\_ID + "=?", new String[] { String.valueOf(id) }, null, null, null); est l'équivalent de "SELECT KEY\_ID, KEY\_NAME, KEY\_PH\_NO FROM TABLE\_CONTACTS WHERE KEY\_ID=' + id + "'"

# rawQuery() : requête SELECT pour Android

- La méthode rawQuery() de SQLiteDatabase permet de lancer des simples requêtes SELECT comme SELECT \* FROM " + TABLE\_CONTACTS WHERE condition paramétrée
- Sa signature est public Cursor rawQuery (String sql, String[] selectionArgs) où sql est une requête SELECT (qui ne doit pas être terminée par ;) et selectionArgs est le tableau qui fixe les valeurs des paramètres (noté ?) dans la clause WHERE
- Par exemple :

```
String countQuery = "SELECT * FROM Etudiants" ;  
SQLiteDatabase db = baseHelper.getReadableDatabase();  
Cursor cursor = db.rawQuery(countQuery, null);
```

# L'objet Cursor (1/2)

- La méthode `query()` retourne un `android.database.Cursor` (~ `java.sql.ResultSet`).`android.database.Cursor` est une interface. Ce qui est retourné est un objet d'une classe qui implémente cette interface
- C'est similaire à JDBC. Le `Cursor` représente un ensemble de "lignes" contenant le résultat de la requête `SELECT`
- `public int getCount()` retourne le nombre de lignes contenues dans le `Cursor`
- On se positionne au début du `Cursor` (= avant la première ligne) par la méthode `public boolean moveToFirst()` (qui retourne `false` si le `Cursor` est vide)
- On teste si on a une nouvelle ligne à lire par la méthode `public boolean moveToNext()` (qui retourne `false` si on était positionné après la dernière ligne)

# L'objet Cursor (2/2)

- On récupère la `columnIndex` cellule de la ligne par la méthode : `public XXX getXXX(int columnIndex)`. `columnIndex` est (évidemment) le numéro de la cellule dans la requête. `XXX` est le type retourné (`String, short, int, long, float, double`)
- Il n'y a pas de `getXXX(String nomDeColonne)` contrairement à JDBC
- On referme le `Cursor` (et libère ainsi les ressources) par `public void close()`
- On peut avoir des renseignements sur le résultat de la requête `SELECT (* FROM ...)` (Méta données) à l'aide du `Cursor` comme :
  - `public int getColumnCount()` qui retourne le nombre de colonnes contenues dans le `Cursor`
  - `public String getColumnName(int columnIndex)` qui retourne le nom de la `columnIndex` ième colonne

# L'accès aux données : un DAO

- Manipuler le Cursor, c'est bien. C'est "un peu" de la programmation "bas niveau"
- Bref un DAO (= Data Access Object), voire une façade s'impose !
- Pour accéder aux données, on masque les bidouilles sous jacentes (requête SQL, etc.) par un objet d'une classe DAO : un bon design pattern !
- Les bidouilles SQL masquées par le DAO sont :
  - insérer des données dans la BD par la méthode `insert()` de la classe `SQLiteDatabase`
  - retourner toutes les données d'une table par la méthode `query()` de la classe `SQLiteDatabase`

# Le code du DAO (1/5)

```
11 public class Etudiants.DAO {  
12     private SQLiteDatabase maBaseDonnees;  
13     private EtudiantOpenHelper baseHelper;  
14     String requeteCreationTable = "Create table Etudiants(id integer primary key autoincrement,  
15             + nom text, prenom text, filiere text);";  
16  
17     public Etudiants.DAO(Context context, String nomBD) {  
18         baseHelper = new EtudiantOpenHelper(context, nomBD, null, 1);  
19     }  
20  
21     public SQLiteDatabase ouvrirBD() {  
22         maBaseDonnees = baseHelper.getWritableDatabase();  
23         Log.i("BD Etudiants", "BD Etudiants crée avec Succés");  
24         return maBaseDonnees;  
25     }  
26  
27     public void fermerBD() {  
28         Log.i("BD Etudiants", "Fermeture de la BD avec Succés");  
29         maBaseDonnees.close();  
30     }  
31 }
```

# Le code du DAO (2/5)

```
32  public Etudiant getEtudiant(int id) {  
33      Cursor c = maBaseDonnees.query("Etudiants", new String[] { "id", "nom", "prenom",  
34          "filiere" }, null, null, null, " id LIKE " + id, null);  
35      Log.i("BD Etudiants", "Etudiant n°" + id + " récupéré avec Succés");  
36      return cursorToEtudiant(c);  
37  }  
38  
39  private Etudiant cursorToEtudiant(Cursor c) {  
40      // Si la requête ne renvoie pas de résultat  
41      if (c.getCount() == 0)  
42          return null;  
43      Etudiant etu = new Etudiant(c.getInt(0), c.getString(1),  
44          c.getString(2), c.getString(3));  
45      // Ferme le curseur pour libérer les ressources  
46      c.close();  
47      return etu;  
48  }
```

# Le code du DAO (3/5)

```
50+ public ArrayList<Etudiant> getAllEtudiants() {  
51     Cursor c = maBaseDonnees.query("Etudiants", new String[] { "id", "nom", "prenom",  
52             "filiere" }, null, null, null, null);  
53     return cursorToEtudiants(c);  
54 }  
55  
56+ private ArrayList<Etudiant> cursorToEtudiants(Cursor c) {  
57     // Si la requête ne renvoie pas de résultat  
58     if (c.getCount() == 0)  
         return new ArrayList<Etudiant>(0);  
59     ArrayList<Etudiant> retEtudiants = new ArrayList<Etudiant>(c.getCount());  
60     c.moveToFirst();  
61     do {  
62         Etudiant etudiant = new Etudiant(c.getInt(0), c.getString(1),  
63                 c.getString(2), c.getString(3));  
64         retEtudiants.add(etudiant);  
65     } while (c.moveToNext());  
66     // Ferme le curseur pour libérer les ressources  
67     c.close();  
68     return retEtudiants;  
69 }  
70 }
```

# Le code du DAO (4/5)

```
72  public long insertEtudiant(Etudiant etudiant) {  
73      ContentValues valeurs = new ContentValues();  
74      valeurs.put("nom", etudiant.getNom());  
75      valeurs.put("prenom", etudiant.getPrenom());  
76      valeurs.put("filiere", etudiant.getFiliere());  
77      Log.i("BD Etudiants", "Etudiant " + etudiant.toString()  
78          + " inséré avec Succés");  
79      return maBaseDonnees.insert("Etudiants", null, valeurs);  
80  }  
81  
82  public void videLaBase() {  
83      // Dans notre cas, nous supprimons la base et les données pour en  
84      // créer une nouvelle ensuite. Vous pouvez créer une logique de mise  
85      // à jour propre à votre base permettant de garder les données à la  
86      // place.  
87      maBaseDonnees.execSQL("drop table Etudiants;");  
88      // Création de la nouvelle structure.  
89      maBaseDonnees.execSQL(requeteCreationTable);  
90      Log.i("BD Etudiants", "BD vidée et restaurée créée avec Succés");  
91  }
```

# Le code du DAO (5/5)

```
93     public void SupprimerEtudiant(Etudiant etudiant) {  
94         maBaseDonnees.delete("Etudiants", "id = ?",
95                             new String[] { String.valueOf(etudiant.getId()) });
96         maBaseDonnees.close();
97         Log.i("BD Etudiants", "Etudiant " + etudiant.toString()
98               + " supprimé avec Succés");
99     }  
100  
101    public void updateEtudiant(int id, Etudiant newInfosEtudiant) {
102        ContentValues values = new ContentValues();
103        values.put("nom", newInfosEtudiant.getNom());
104        values.put("prenom", newInfosEtudiant.getPrenom());
105        values.put("filiere", newInfosEtudiant.getPrenom());
106        maBaseDonnees.update("Etudiants", values, "id = ?",
107                             new String[] { String.valueOf(newInfosEtudiant.getId()) });
108    }  
109 }
```

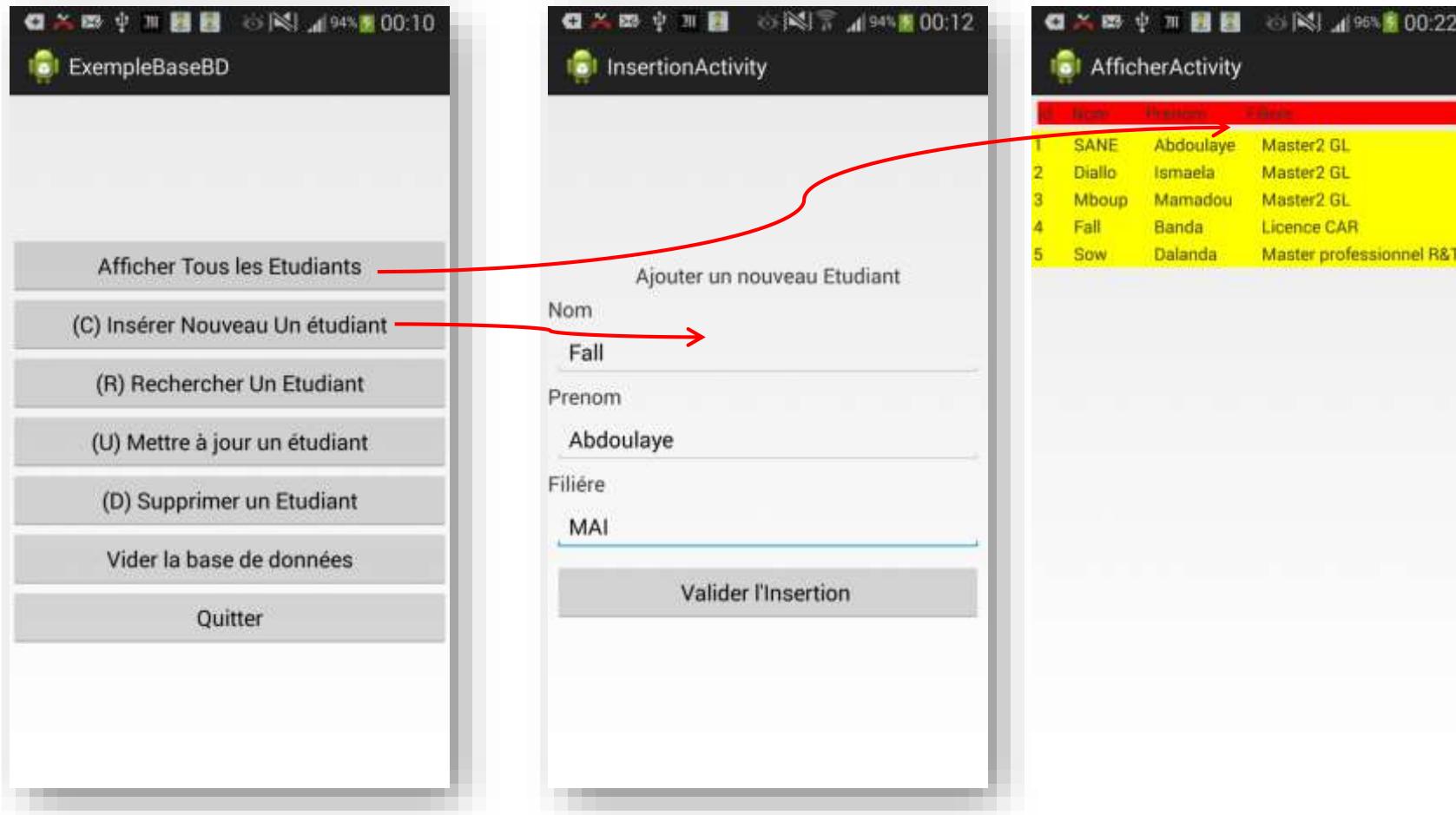
# A propos de Supprimer des lignes dans une table :DELETE

- Pour supprimer des lignes dans une table, la méthode utilisée (de la classe SQLiteDatabase) est `public int delete (String table, String whereClause, String[] whereArgs)`
  - `table` est la table à manipuler
  - `whereClause` est la clause WHERE filtrant les lignes à supprimer. Si la valeur est null, toutes les lignes sont détruites
  - `whereArgs` indiquent les valeurs à passer aux différents arguments de la clause WHERE qui sont notés ? dans `whereClause`
- Cette méthode retourne le nombre de ligne qui ont été supprimées
- Exemple : Voir code du DAO

# Modifier une table : UPDATE de SQL pour Android

- Pour mettre à jour des lignes dans une table, la méthode utilisée (de la classe `SQLiteDatabase`) est `public int update (String table, ContentValues values, String whereClause, String[] whereArgs)` où :
  - `table` est la table qui doit être mise à jour
  - `values` est une suite de couples (clé, valeur) où la clé, de classe `String`, est le nom de la colonne et valeur, sa valeur
  - `whereClause` est la clause WHERE filtrant les lignes à mettre à jour. Si la valeur est null, toutes les lignes sont mises à jour
  - `whereArgs` indiquent les valeurs à passer aux différents arguments de la clause WHERE qui sont notés ? dans `whereClause`
  - Cette méthode retourne le nombre de ligne qui ont été affectées
- Exemple : Voir DAO

# Exemple à compléter...



# Et pour le reste ???

- Les capteurs
- Les services
- La géolocalisation
- ...