DÉVELOPPEMENT D'APPLICATIONS MOBILES

PARTIE 3 - CAPACITOR ET APP HYBRIDES

COURS 1 - INTRODUCTION À CAPACITOR

ALEX MOREL

CODE LUTIN

@alex_morel_

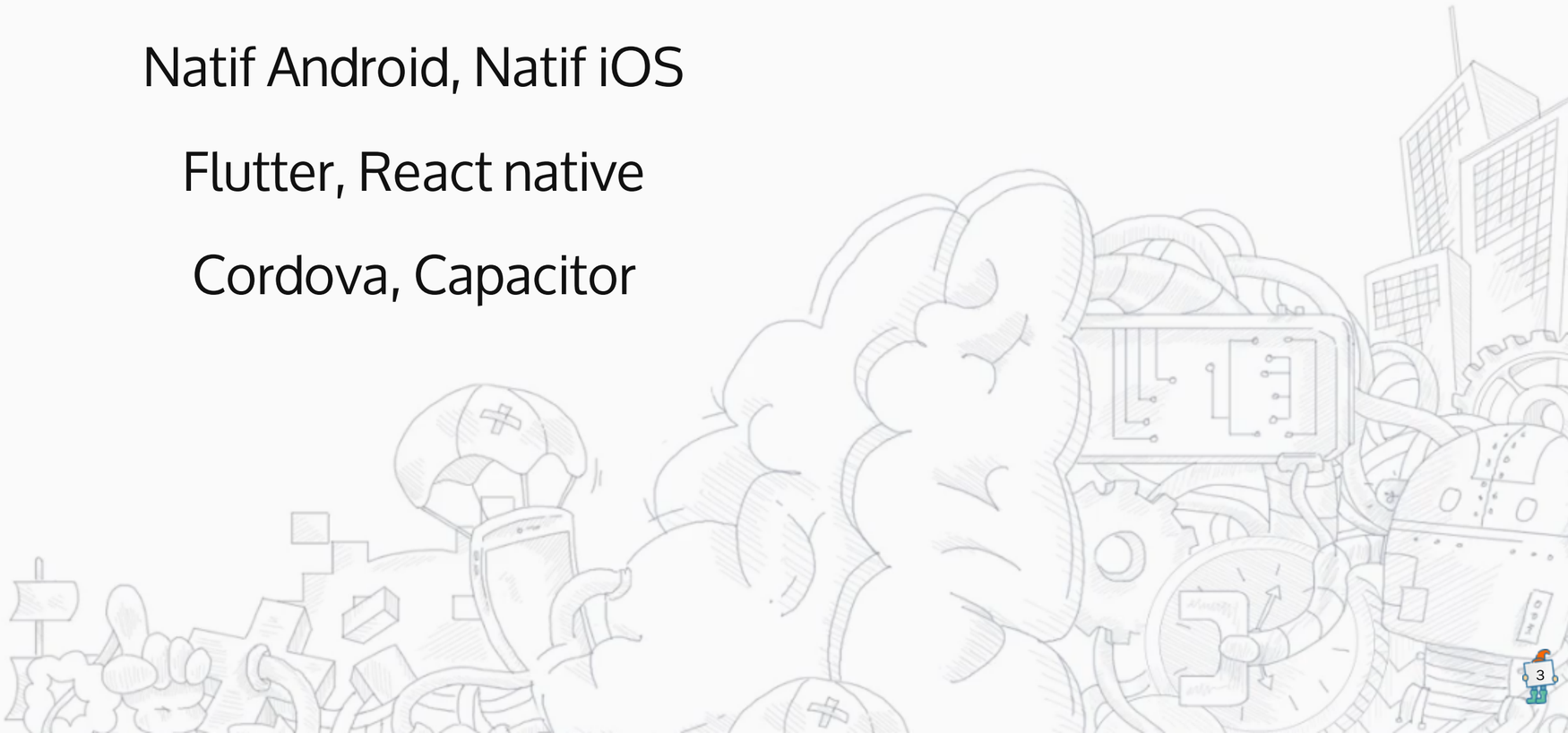
1. HYBRIDE, COMPILÉ, NATIF ?

De nombreuses technos permettent d'obtenir des applications mobiles (apk sur les stores). Lesquelles ?

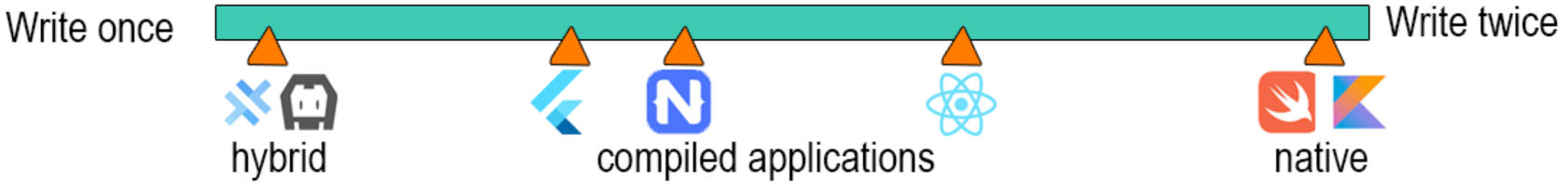
Natif Android, Natif iOS

Flutter, React native

Cordova, Capacitor

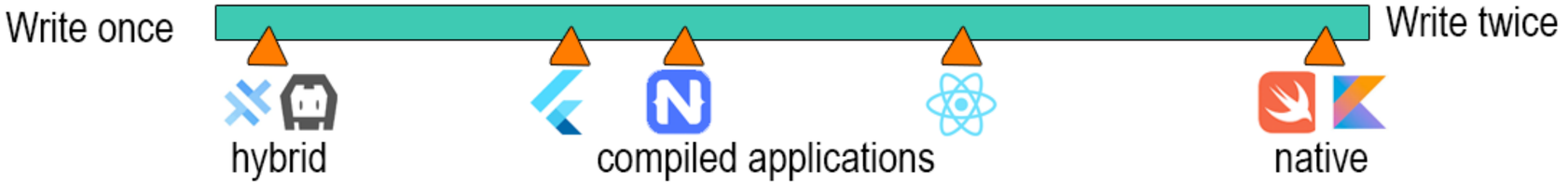


«Write once, use everywhere»



«Learn once, write everywhere»

«Write once, use everywhere»



«Learn once, write everywhere»

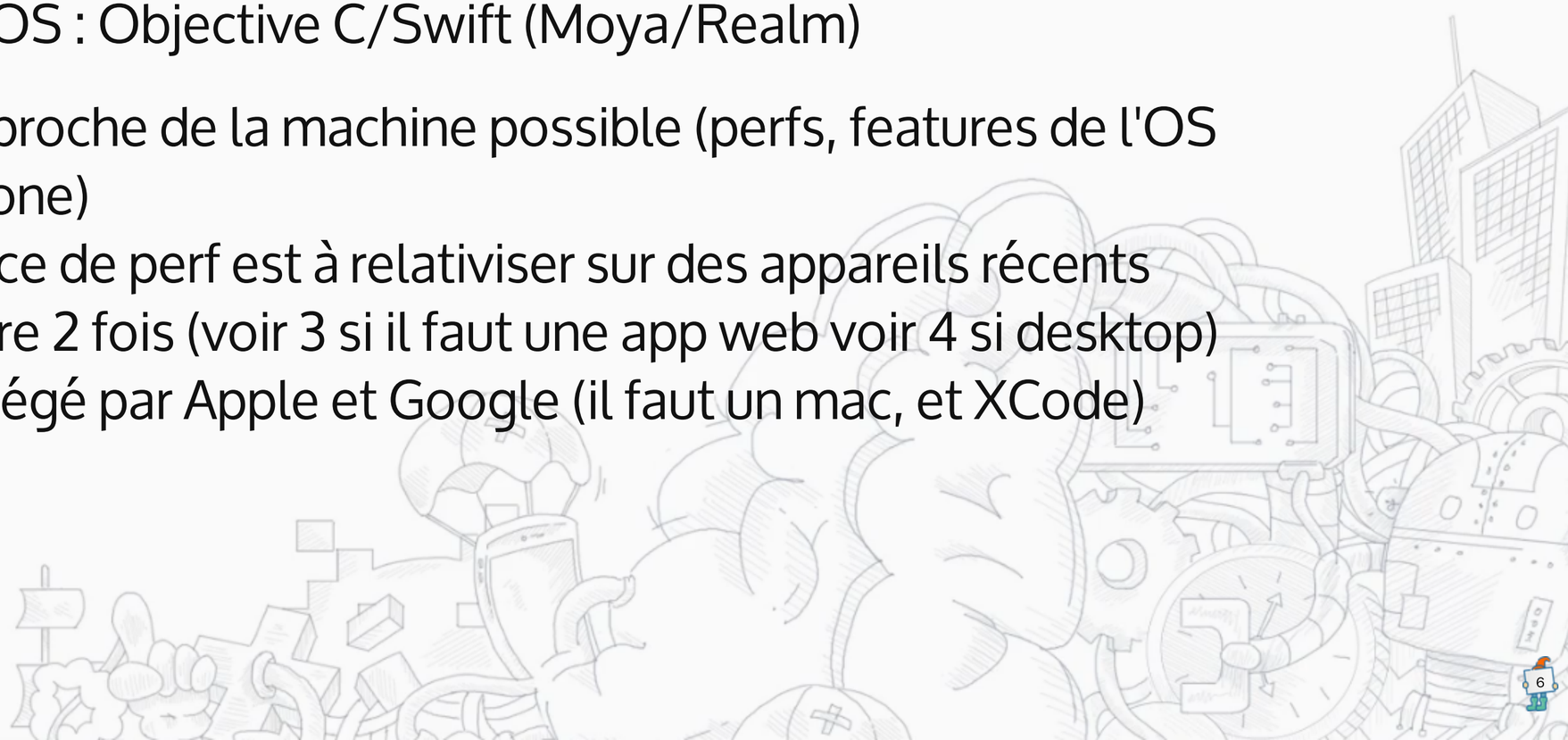
On va distinguer 4 familles de solutions:
Natif, Semi-natif, Compilé, Full Hybride

APPLICATION NATIVES

Android : Java/Kotlin (AAC, Retrofit, Room)

iOS : Objective C/Swift (Moya/Realm)

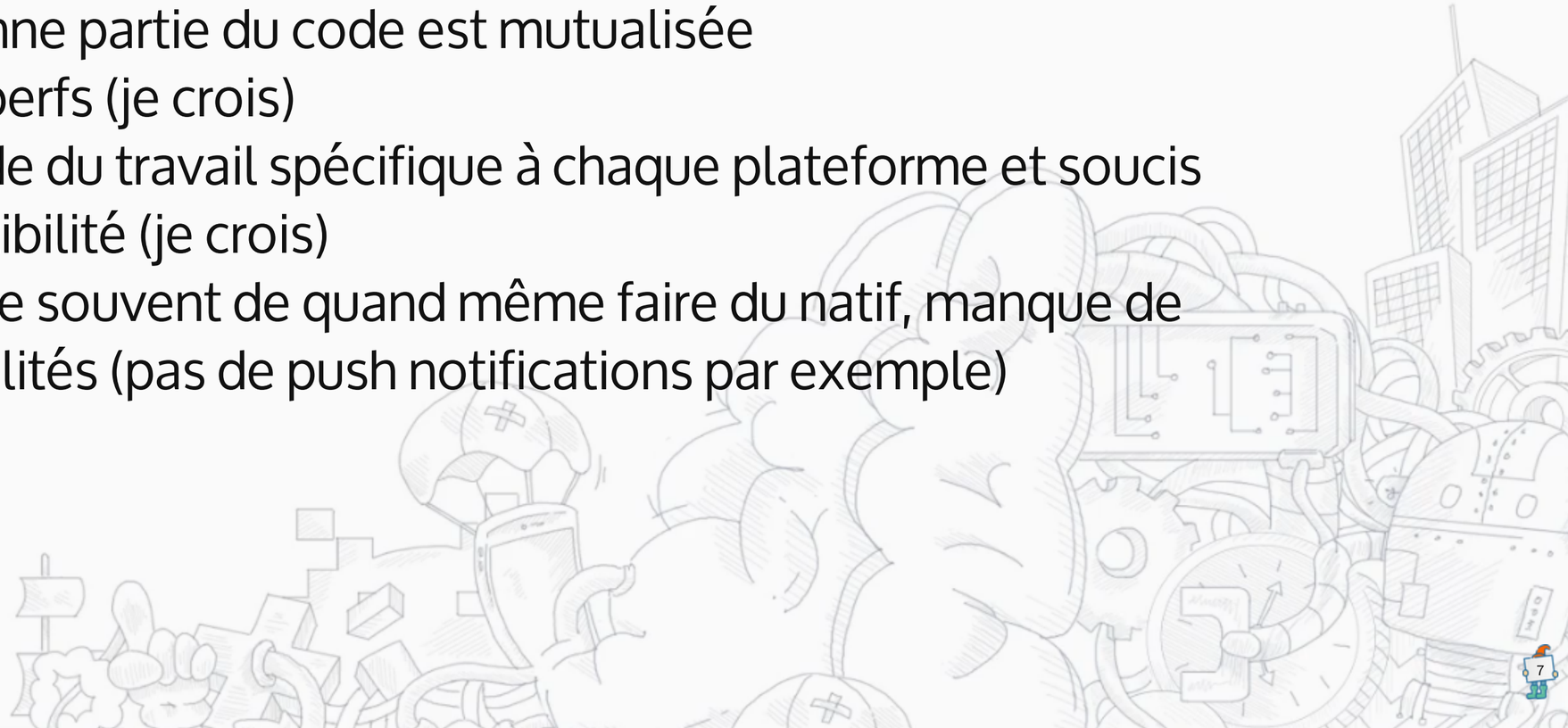
- + : Le plus proche de la machine possible (perfs, features de l'OS dispo day one)
- La différence de perf est à relativiser sur des appareils récents
- - : tout écrire 2 fois (voir 3 si il faut une app web voir 4 si desktop)
- - : on est piégé par Apple et Google (il faut un mac, et XCode)



APPLICATION SEMI-NATIVES

React Native: logique en React, mais appelle l'UI native Android/iOS

- + : Une bonne partie du code est mutualisée
- + : Bonne perfs (je crois)
- - : Demande du travail spécifique à chaque plateforme et soucis de compatibilité (je crois)
- - : demande souvent de quand même faire du natif, manque de fonctionnalités (pas de push notifications par exemple)

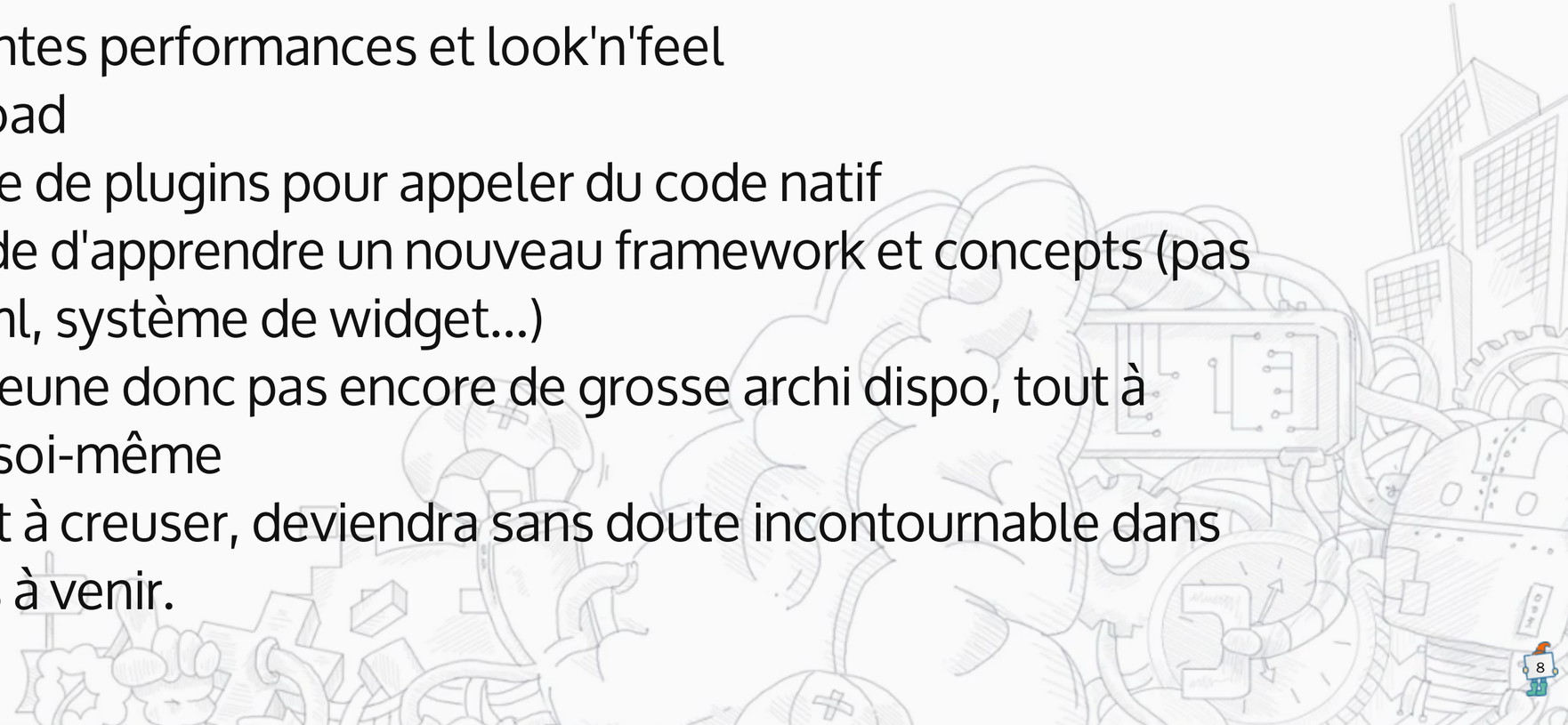


APPLICATION COMPILÉES

Unity: moteur de jeu (C# + framework dédié), compilé vers Android,iOS...

Flutter: code en Dart, compilé vers Android, iOS (et Linux + Web depuis 2.0)

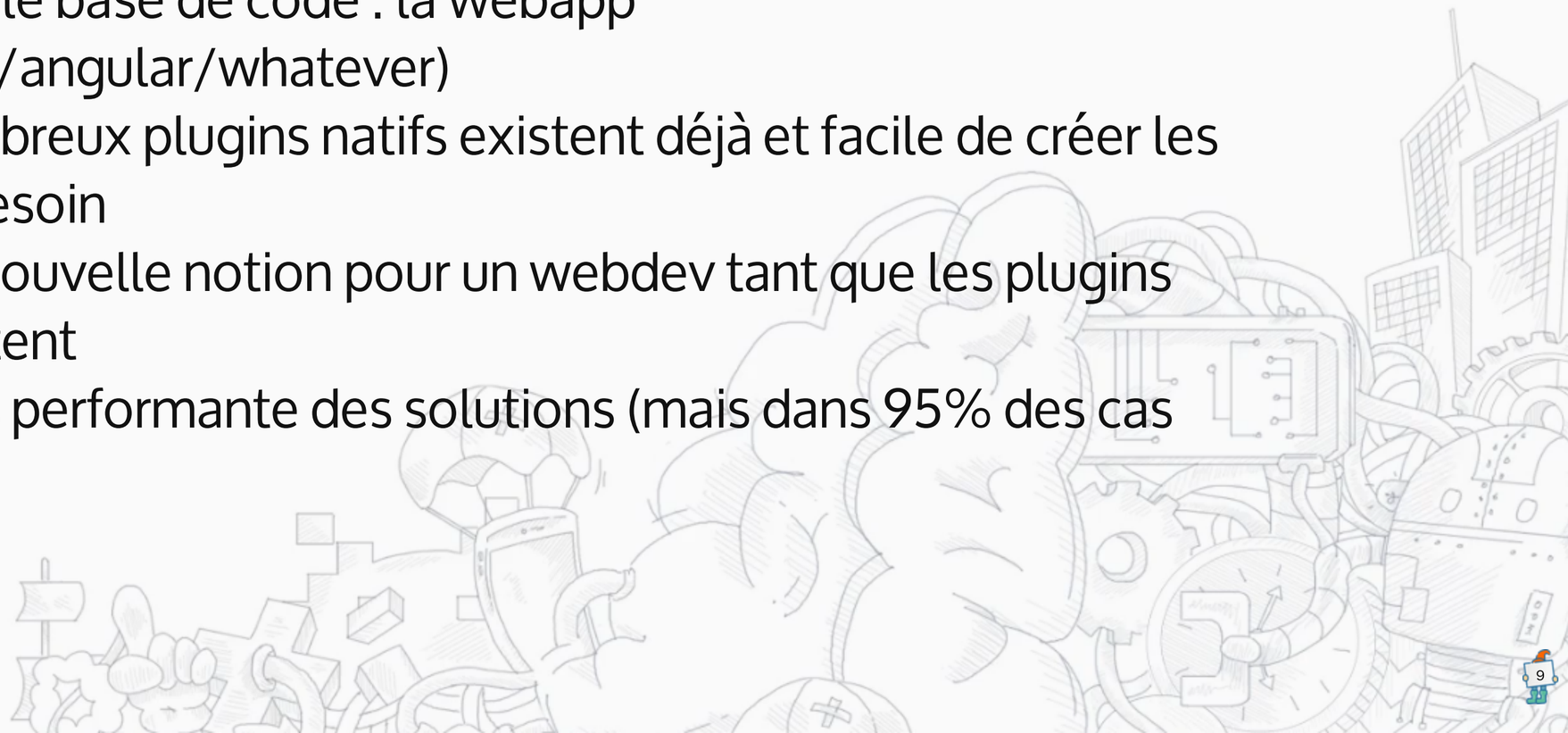
- + : Une seule base de code
- + : Excellentes performances et look'n'feel
- + : Hot reload
- + : Système de plugins pour appeler du code natif
- - : Demande d'apprendre un nouveau framework et concepts (pas de css, html, système de widget...)
- - : Un peu jeune donc pas encore de grosse archi dispo, tout à structurer soi-même
- Clairement à creuser, deviendra sans doute incontournable dans les années à venir.



APPLICATION FULL HYBRIDES

Anciennement Cordova, maintenant Capacitor: application JS qui envoie des messages à des plugins natifs.

- + : Une seule base de code : la webapp (vue/react/angular/whatever)
- + : De nombreux plugins natifs existent déjà et facile de créer les votre au besoin
- + : Aucun nouvelle notion pour un webdev tant que les plugins natifs existent
- - : la moins performante des solutions (mais dans 95% des cas OSEF)



CAPACITOR EN 2 MOTS

Capacitor génère des projets XCode / AndroidStudio / Electron à partir d'une webapp. Ces projets sont natifs, ils se buildent et se configurent comme des projets natifs (e.g. AndroidManifest.xml, build.gradle...)

Le projet généré est une "Super WebView" qui va embarquer la webapp et fournir des API js pour appeler des plugins natifs (GPS, bluetooth, vibreur...)

Le crédo de Capacitor : **"Write once, run Everywhere"**

2. FROM WEB TO MOBILE

Ecrivez votre webapp comme d'habitude (Angular/React/Vue/Js pur), puis ensuite:

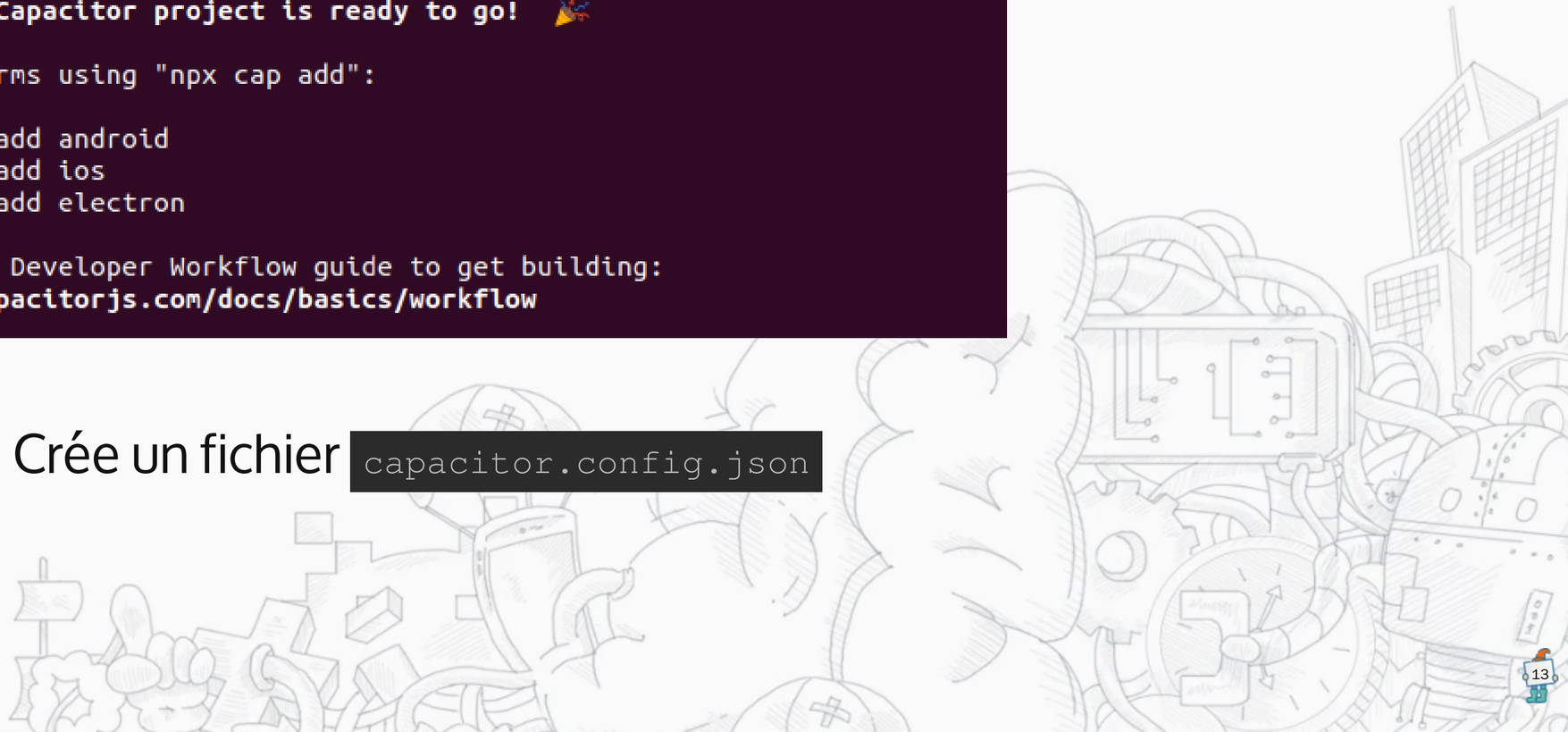
```
npm install @capacitor/core @capacitor/cli
```

Installe les libraires Capacitor ainsi que la CLI (`npx cap *`) dans votre webapp existante

```
npx cap init
```

```
> npx cap init
? App name compostmap
? App Package ID (in Java package format, no dashes) org.univ.compostmap
✔ Initializing Capacitor project in /home/alexsedlex/Documents/blog/org.ur


🎉   Your Capacitor project is ready to go!  🎉

Add platforms using "npx cap add":

  npx cap add android
  npx cap add ios
  npx cap add electron

Follow the Developer Workflow guide to get building:
https://capacitorjs.com/docs/basics/workflow
```

Crée un fichier `capacitor.config.json`



Crée un fichier `capacitor.config.json`

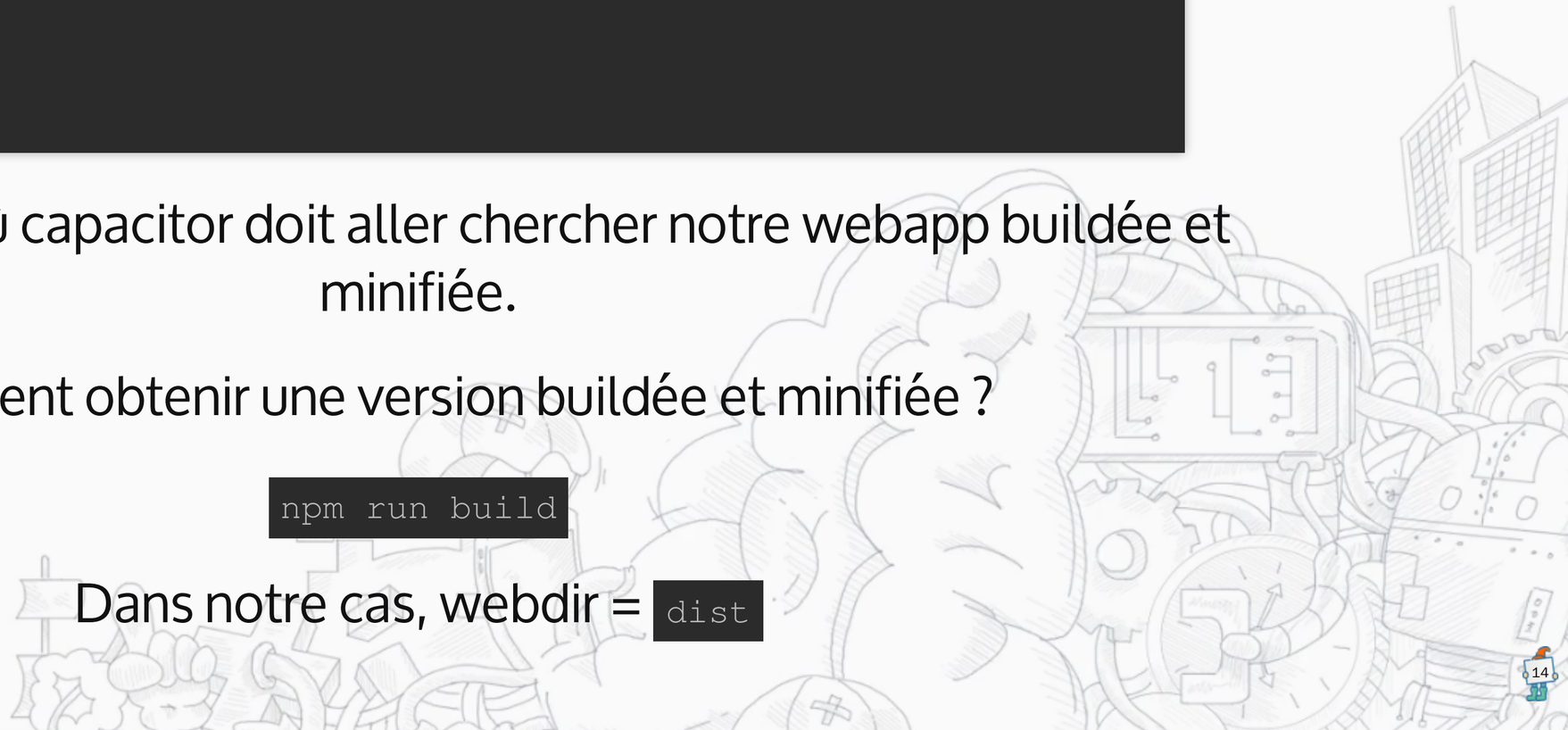
```
{
  "appId": "org.univ.compostmap",
  "appName": "compostmap",
  "bundledWebRuntime": false,
  "npmClient": "npm",
  "webDir": "dist",
  "plugins": {
    "SplashScreen": {
      "launchShowDuration": 0
    }
  },
  "cordova": {}
}
```

`webdir` indique où capacitor doit aller chercher notre webapp buildée et minifiée.

Comment obtenir une version buildée et minifiée ?

```
npm run build
```

Dans notre cas, webdir = `dist`



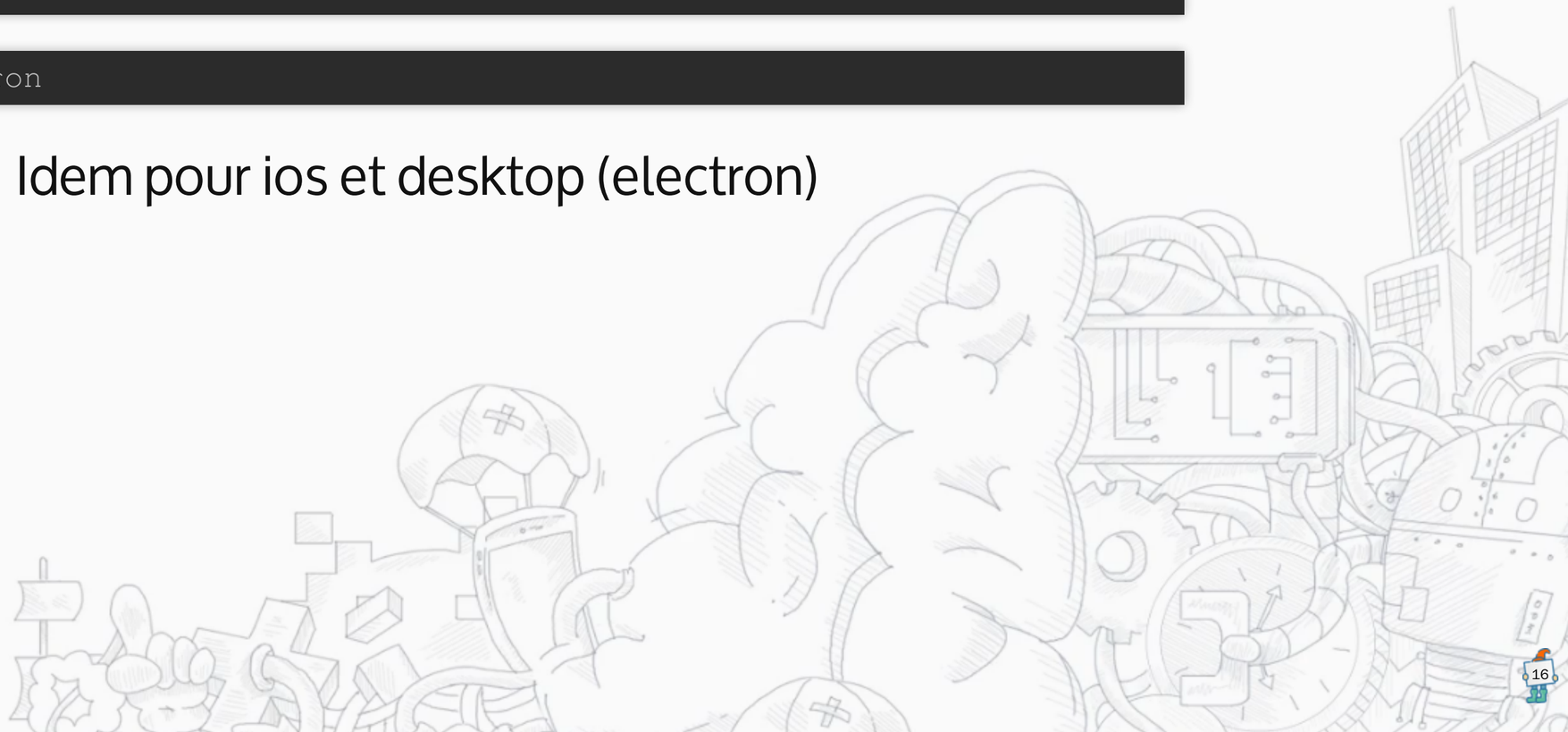
```
npx cap add android
```

Génére un vrai projet Android qu'on peut ouvrir avec android studio

```
npx cap add ios
```

```
npx cap add electron
```

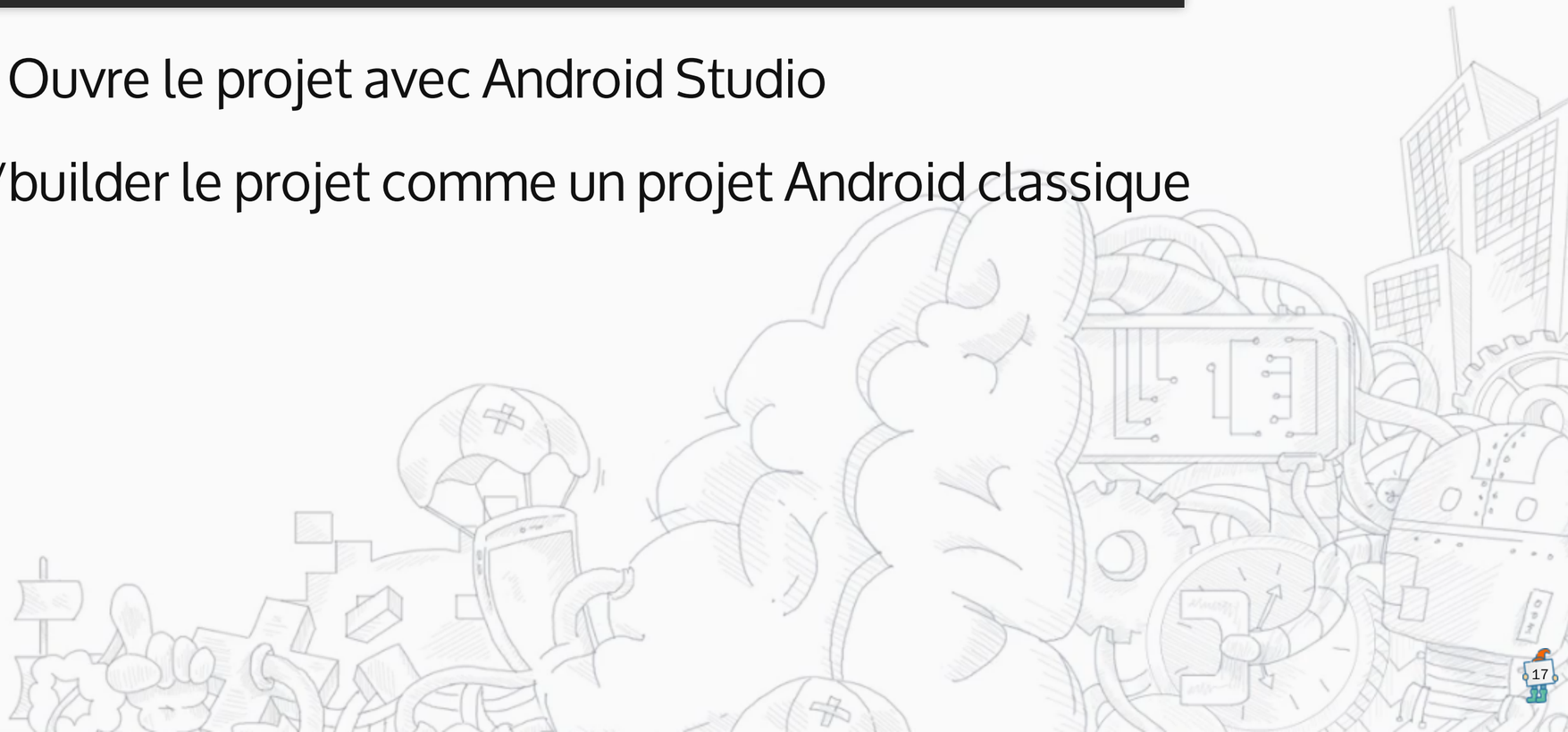
Idem pour ios et desktop (electron)



```
npx cap open android
```

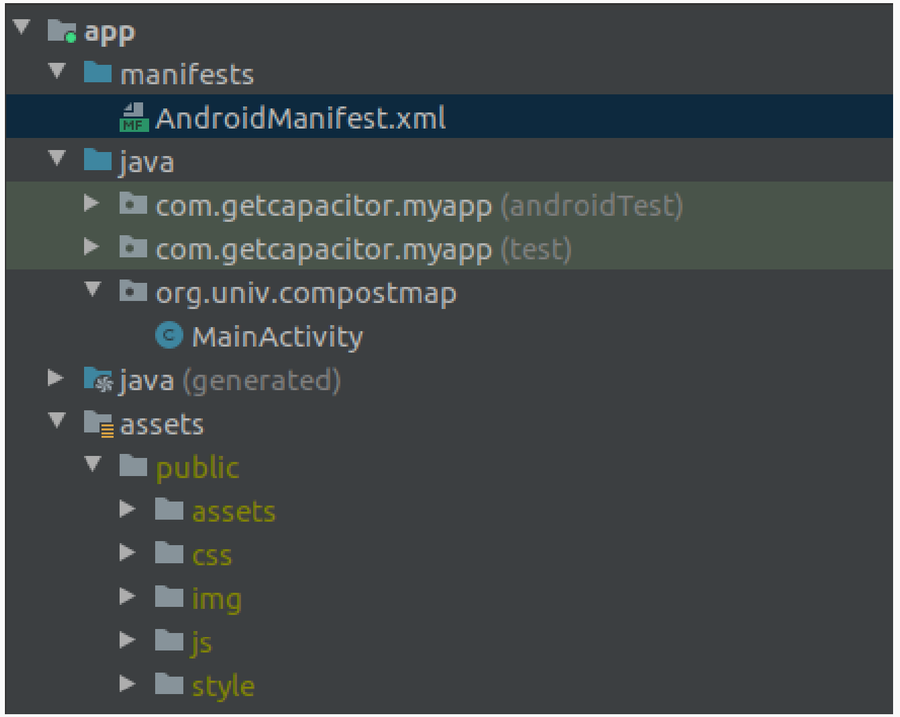
Ouvre le projet avec Android Studio

On peut lancer/builder le projet comme un projet Android classique



EXERCICE 20: COMPOSTMAP SUR ANDROID

Utilisez Capacitor pour créer la version mobile de Compostmap



On retrouve notre bon vieux manifest (les permissions c'est toujours là que ça se passe)

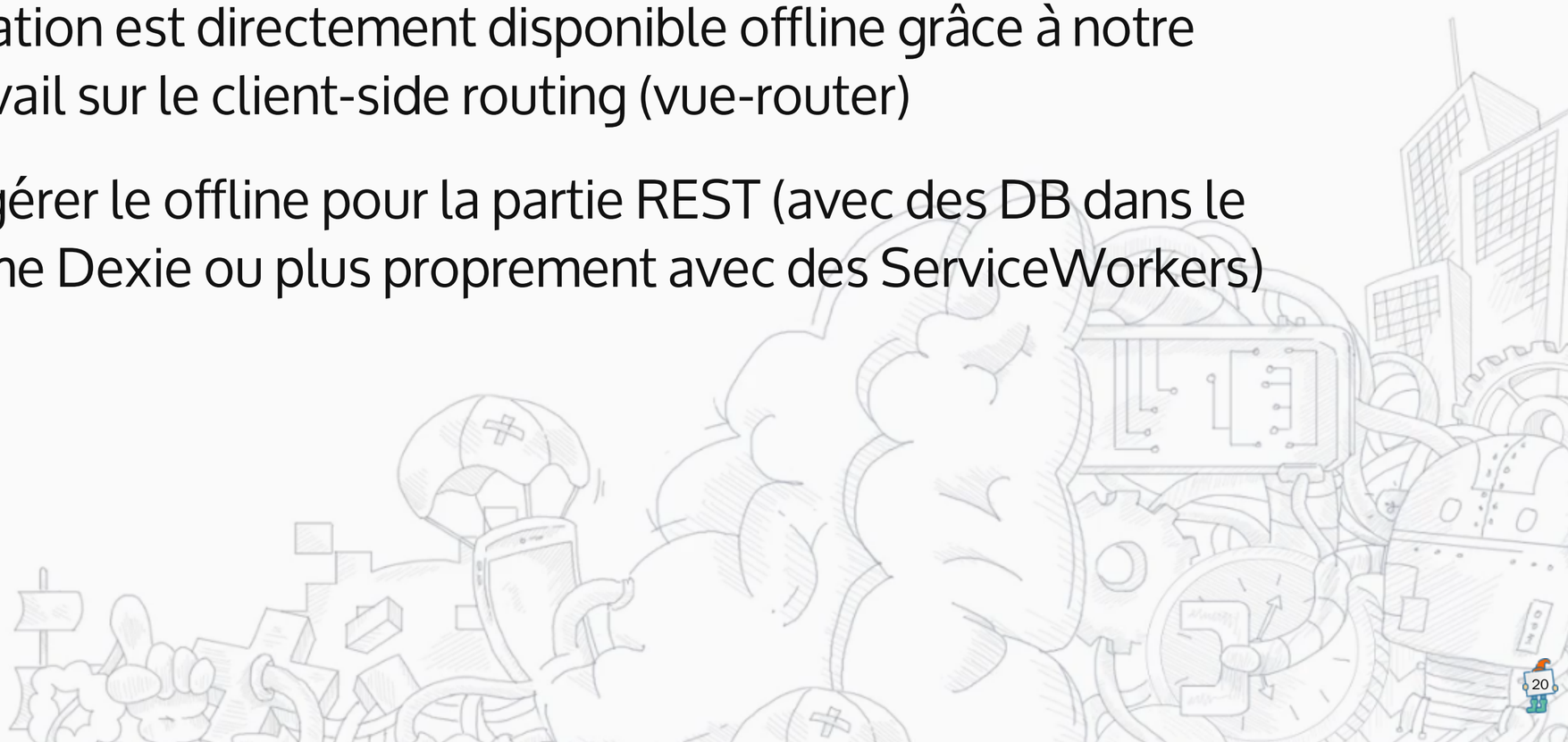
Unique activité `MainActivity`, la "super WebView" (on reviendra dessus)

Notre webapp Vue (dossier dist) est embarquée dans les assets

Notre webapp Vue (dossier dist) est embarquée dans les assets

Et donc l'application est directement disponible offline grâce à notre travail sur le client-side routing (vue-router)

Bien sûr il faudra gérer le offline pour la partie REST (avec des DB dans le localstorage comme Dexie ou plus proprement avec des ServiceWorkers)



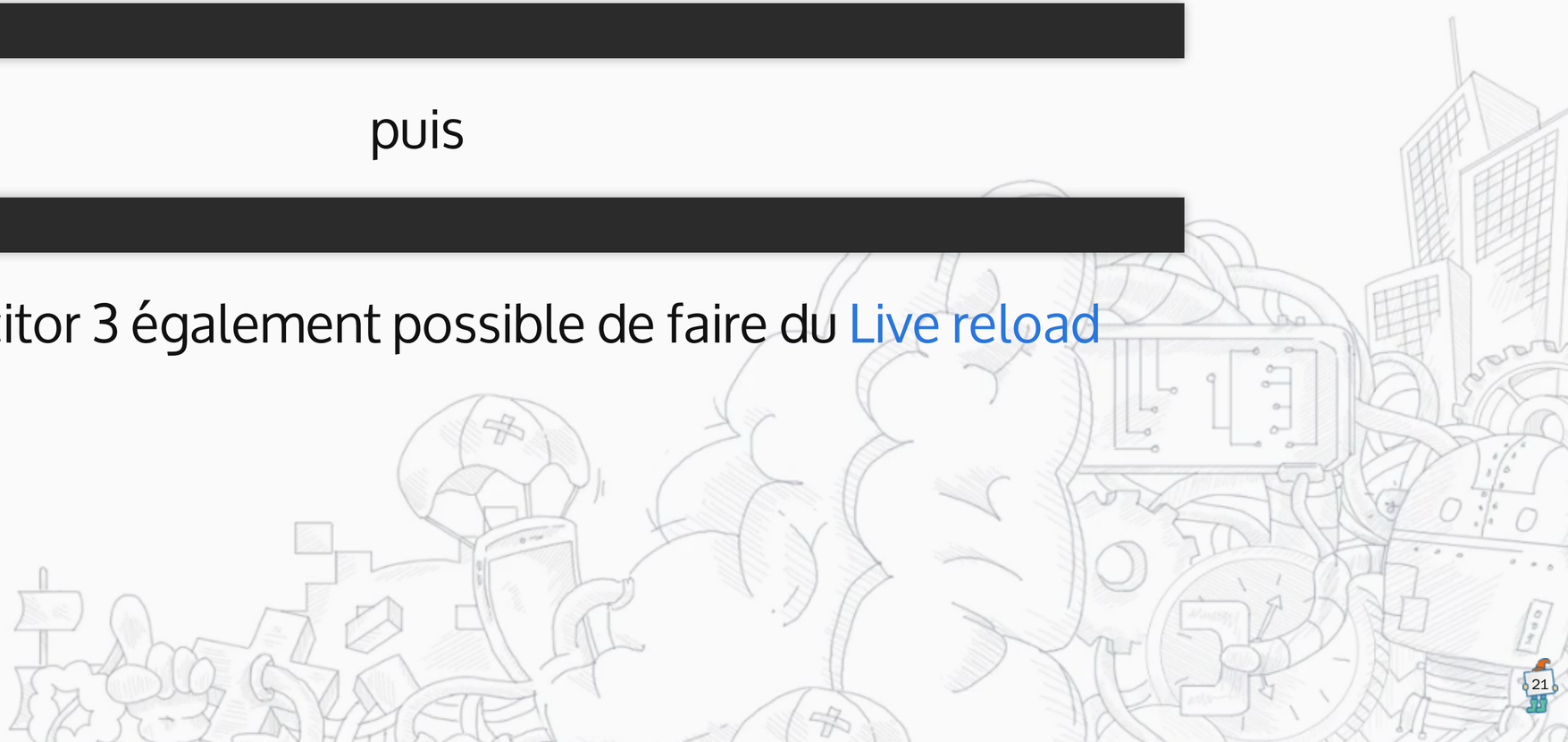
Pour mettre à jour l'application mobile quand on fait un changement dans la webapp :

```
npm run build
```

puis

```
npx cap copy
```

Avec Capacitor 3 également possible de faire du Live reload



Il est possible d'automatiser le `npx cap copy` et la création de l'APK signé dans la CI (exemple gitlab Android)

L'APK signé est directement publiable sur le PlayStore (il faut un compte payant, je peux prêter le mien)

3. NATIVE WHERE IT MATTERS

La philosophie de capacitor: du JS (Vue/React/Angular) partout où on peut

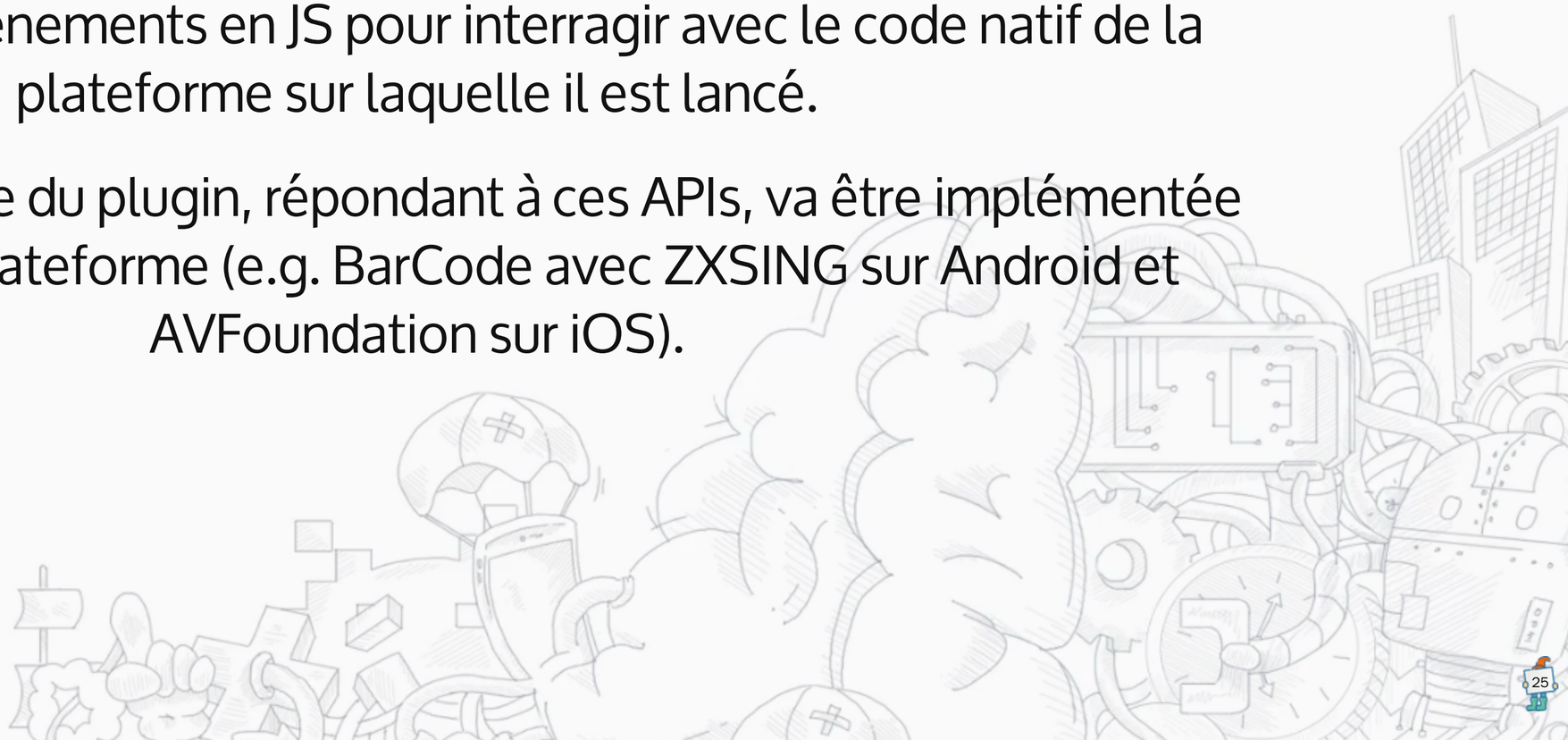
Du natif où c'est vraiment nécessaire. Des exemples ?

Haptics (vibreur), Accéléromètre, Appareil photo, Réalité augmentée...

Pour le natif, Capacitor repose sur la notion de plugin.

Chaque plugin va définir des API en JS. La "super webview" Capacitor va envoyer des évènements en JS pour interragir avec le code natif de la plateforme sur laquelle il est lancé.

Une version native du plugin, répondant à ces APIs, va être implémentée pour chaque plateforme (e.g. BarCode avec ZXSING sur Android et AVFoundation sur iOS).



Exemple : appareil photo (avec permissions)

```
import { Plugins, CameraResultType } from '@capacitor/core';
const { Camera } = Plugins;

Camera.getPhoto({
  quality: 95,
  allowEditing: false,
  resultType: CameraResultType.DataUrl,
  promptLabelCancel: 'My Cancel label',
  promptLabelPhoto: 'Select picture from gallery',
  promptLabelPicture: 'Take picture with camera'
}).then(
  image => {
    if (image.dataUrl) {
      MyPicturesService.savePicture(image.dataUrl);
    }
  },
  failure => {
    console.error('[Picture] Unable to take picture', failure);
  }
);
```

Exemple : position GPS (avec permissions)

```
import { Plugins, GeolocationPosition, CallbackID } from '@capacitor/core';
const { Geolocation } = Plugins;

// Get position: returns the current location, you won't be getting updates
const coordinates = await Geolocation.getCurrentPosition();

// Watch position: only call this when actually
// needing to listen for position changes
let options = {
    enableHighAccuracy: false,
    maximumAge: 20,
    timeout: 3000
};
let watchId:CallbackID = Geolocation.watchPosition(options, (position, err) => {
  if (error != null && position != null) {
      // Do something with position
  }
});
```

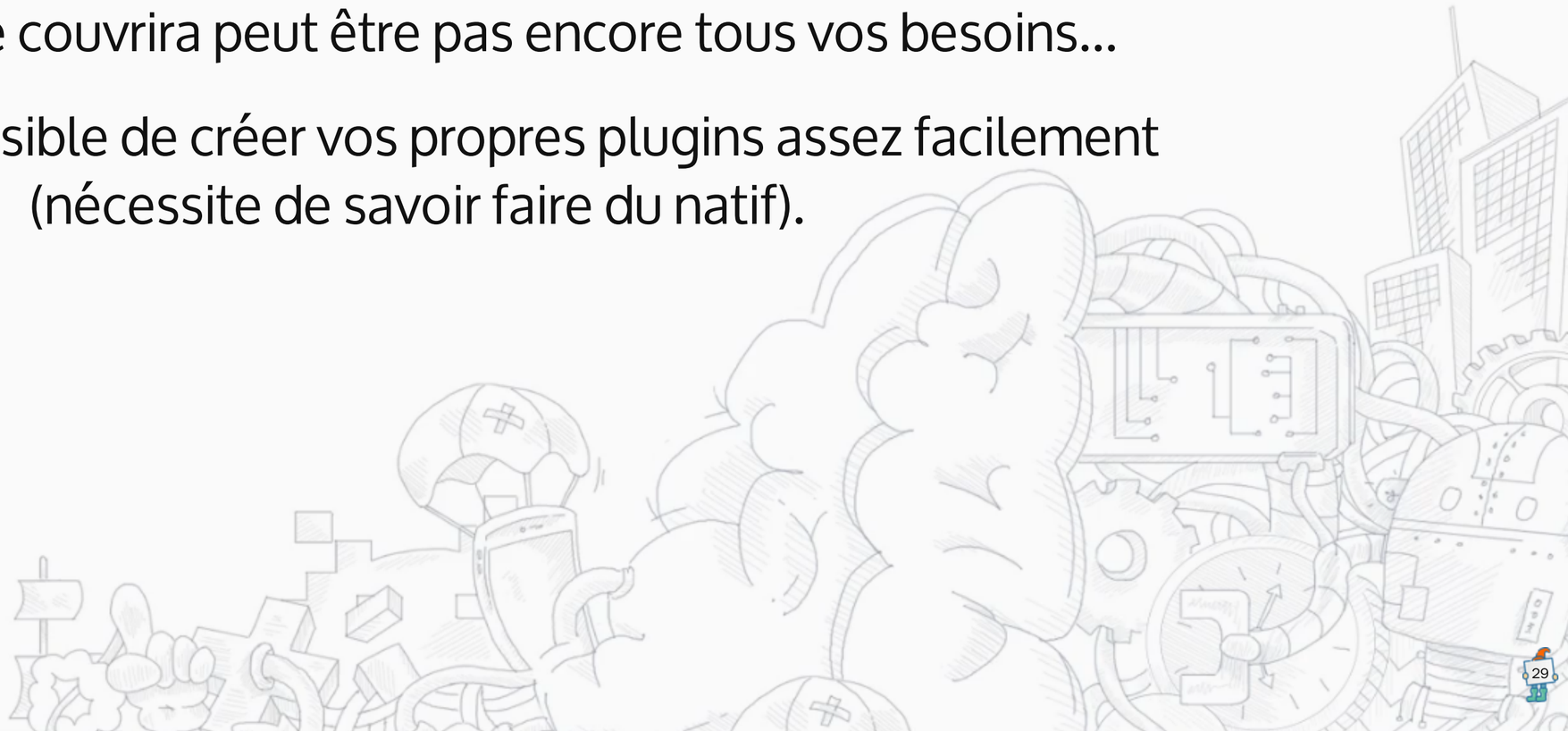
Usage plus complexe : Universal Linking

```
import { Plugins, AppState } from '@capacitor/core';
const { App } = Plugins;

@Component
export default class AppView extends Vue {
  created() {
    // If app is opened from mails when validating account or password forgotten
    App.addListener('appUrlOpen', (data: any) => {
      // Parse date.url and extract action & token
      let token = data.url.substring([...]);
      let action = data.url.substring([...]);
      if ('reset-password' === action) {
        router.push({name:'reset-password', params: {token: token}});
      } else {
        router.push({name:'verify', params: {token: token}});
      }
    });
```

Permet à l'application d'intercepter certaines URL (e.g. `http://compostmap.fr/compost/33`), de se lancer et d'afficher un écran en fonction

Des dizaines de plugins officiels et communautaires sont disponibles.
Capacitor ne couvrira peut être pas encore tous vos besoins...

Mais il est possible de créer vos propres plugins assez facilement
(nécessite de savoir faire du natif).



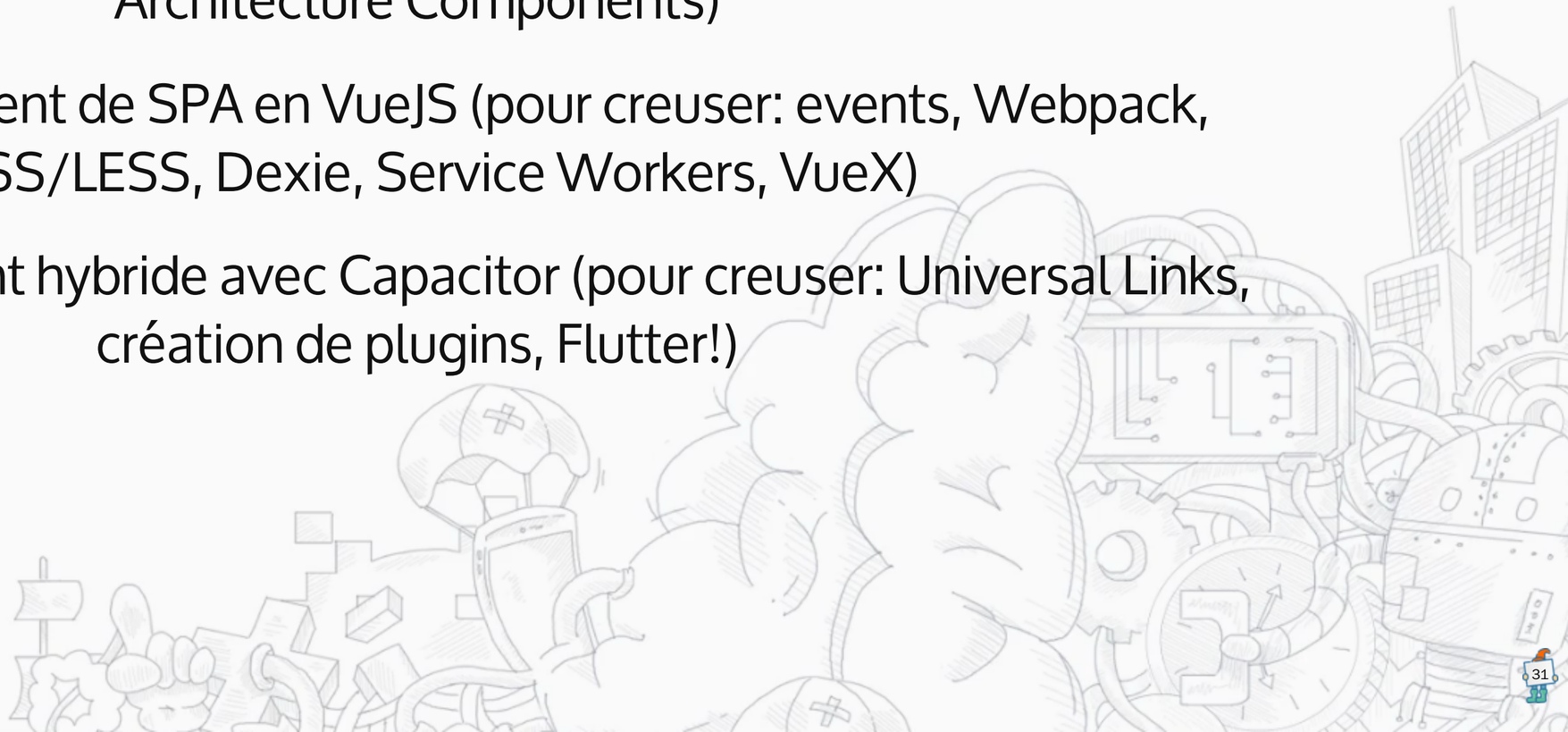
CONCLUSION DU MODULE

Le module nous a permis de nous initier:

Au développement Android Natif (pour creuser: Kotlin, Android Architecture Components)

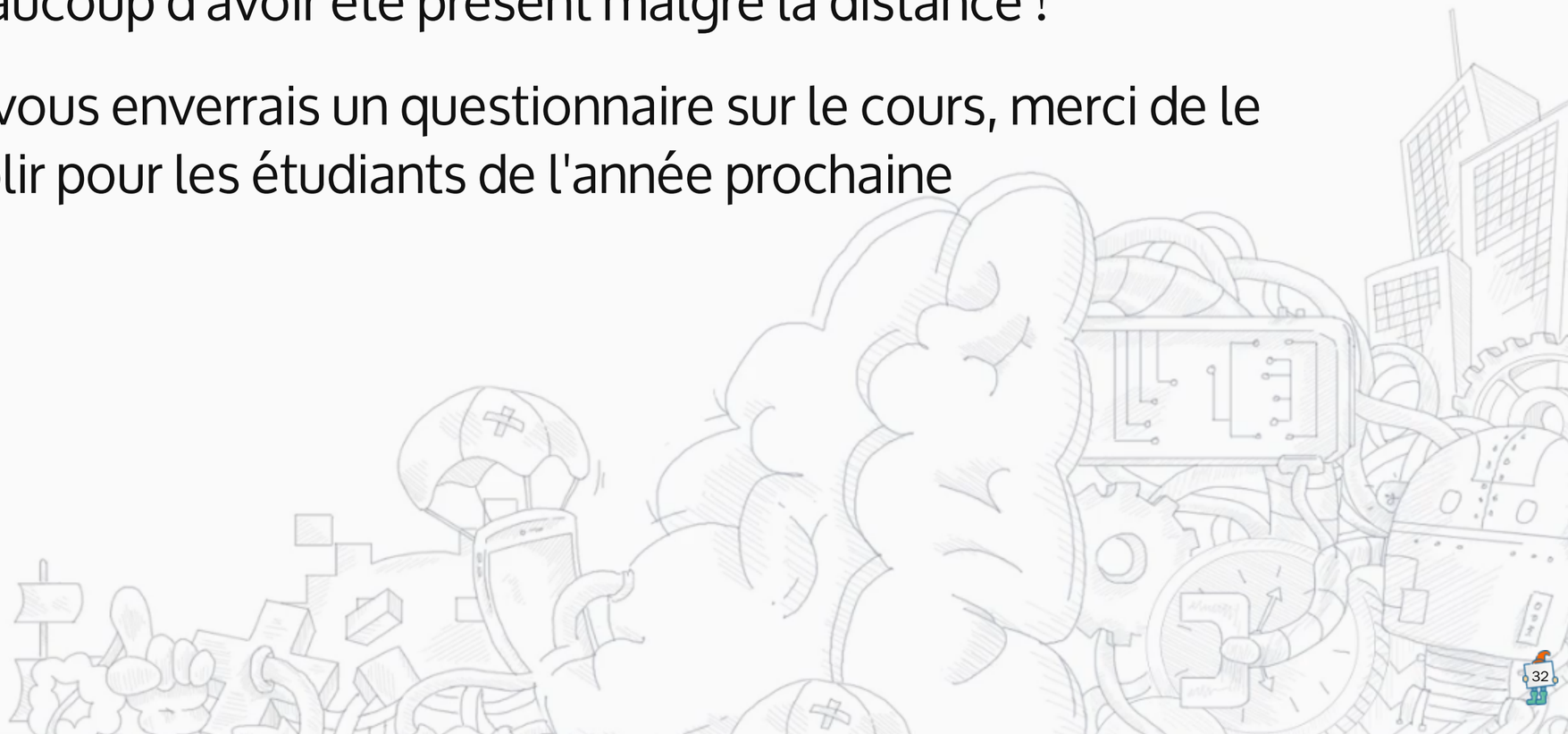
Au développement de SPA en VueJS (pour creuser: events, Webpack, SCSS/LESS, Dexie, Service Workers, VueX)

Au développement hybride avec Capacitor (pour creuser: Universal Links, création de plugins, Flutter!)



Merci beaucoup d'avoir été présent malgré la distance !

Après l'examen je vous enverrais un questionnaire sur le cours, merci de le remplir pour les étudiants de l'année prochaine



MERCI POUR VOTRE ATTENTION !

- Ma porte reste ouverte (par mail et/ou sur discoord et/ou sur twitter)

ALEX MOREL

CODE LUTIN

@alex_morel_