



KUBERNETES

Une introduction de
B. Adanlessossi

AGENDA



DEPLOYMENT DES
MICROSERVICES



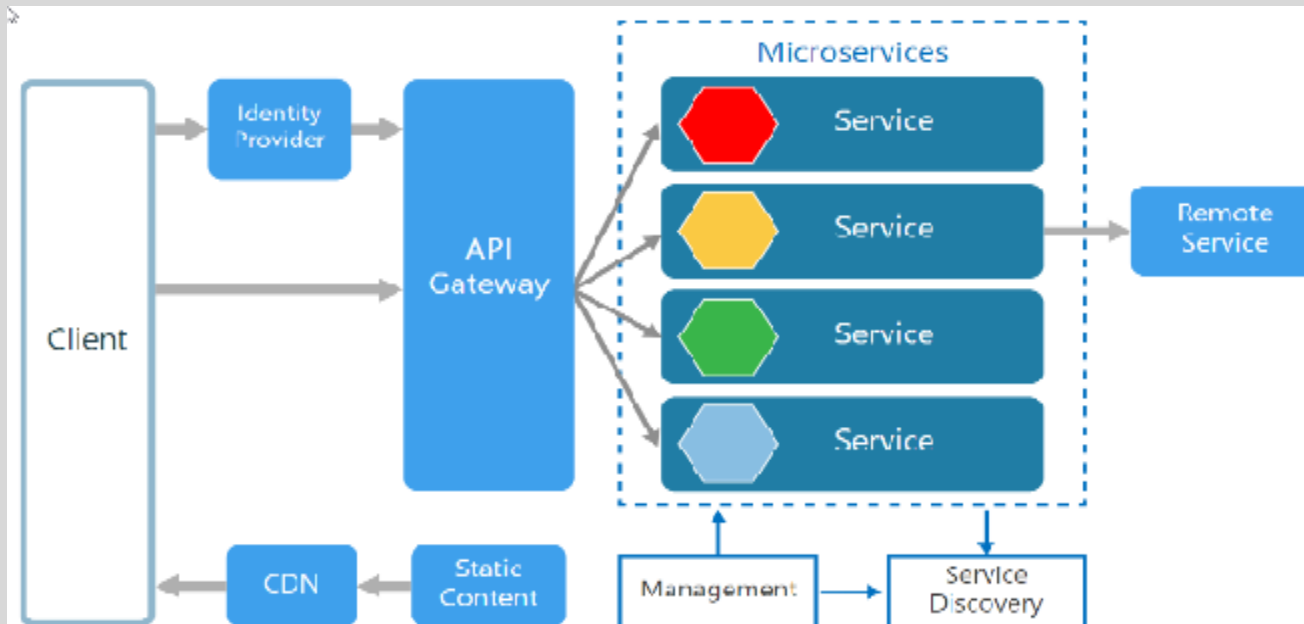
KUBERNETES



DEMO GOOGLE
CLOUD

Les Microservices - rappel

- Dans un précédent workshop, nous avons abondamment parlé des microservices
- Ce sont des services déployés de manière autonome l'un de l'autre mais représentant une plus grande plus value



Déploiement des microservices

- Le déploiement des microservices constitue un véritable challenge.
- L'architecture Microservices est une collection de petits services, chaque service ayant une fonction spécifique. Chacun de ces petits services dispose de sa propre base de données
- Ces petits services sont très performants en isolation mais ont besoin d'un moyen pour interagir entre eux et partager des données.
- Ceci nous amène à une question:
 - Comment ces services peuvent interagir entre elles pour fournir les fonctionnalités de l'application?
- Il existe 2 façons de faire communiquer ces services:
 - - par Chorégraphie
 - - par Orchestration

Chorégraphie

Exemple: animation Populaire au Togo dans les années 70's



Chorégraphie

- La chorégraphie permet de résoudre le problème d'interaction entre les microservices.
- Nous voulons réduire la dépendance entre les services de manière que chaque service puisse fonctionner de manière autonome
- Dans la chorégraphie des microservices, chaque service exécute ses actions de manière autonome et ne requiert aucune instruction extérieure.
- Le service sait à quoi et comment répondre à une interaction



Orchestration

- Dans un Orchestre, nous avons un personnage central appelé “orchestrateur”, “dirigeant”, “chef d'orchestre”, etc...
- Sa fonction principale est d'invoquer les instruments et les sons à jouer à un moment précis.
- Dans l'orchestration des microservices, un chef d'orchestre (controlleur central) gère les interactions entre les microservices. Il transmet les événements aux services et centralise les appels entre les différents services.



Quelle approche adopter?

- Chorégraphie et orchestration sont deux différentes approches pour gérer les microservices.
- L'orchestration utilise une approche centralisée pour exécuter les décisions et permet d'avoir un meilleur contrôle.
- Cependant la chorégraphie donne plus de liberté aux services pour prendre leurs décisions.
- Ces deux approches se valent bien mais laquelle adopter?
- La réponse est une approche hybride dépendant de votre cas d'utilisation.
- ***Mais quels outils avons nous à disposition?***

Docker pour l'orchestration et la chorégraphie

- Docker constitue l'élément principal pour empaquetter les applications.
- Quand on parle de déploiement, on veut dire déploiement de conteneurs.
- Dans nos séances précédentes, nous avons passé pas mal de temps à comprendre les conteneurs docker
- Nous avons eu à créer des images de nos applications.
- Nous avons poussé les images créées sur docker-hub
- Nous étions capable d'exécuter nos images.
- Mais la question revient toujours:
 - **Pourquoi avons nous besoin de docker?**

Pourquoi docker?

Standardisation:

- Standardisation de l'emballage pour tout type d'application

Capacités:

- Langage neutre
- Cloud neutre
- Standardisation

Challenge:

- 1000 Microservices
- 1000 Instances

Docker

Scenario 1

- Afin de mieux comprendre pourquoi docker, imaginons un simple scenario.
- Vous voulez déployer une application. Vous allez vers le responsable pour lui en faire part.
- Il vous demande: Hey, quelle est ton image docker? Vous lui donnez le nom de l'image.
- Il tape la commande
 - `docker run -p 8080:8080 adanlessossi/hello-world-rest-api:0.0.1.RELEASE`
- Et en quelque secondes, votre application s'exécute déjà!
- Ce qui est remarquable, c'est que le responsable ne vous a pas demandé:
 - - Quel framework ton application utilise?
 - - Quel langage de programmation est utilisé par l'application
 - - Sur quel système d'exploitation devra s'exécuter l'application
 - - De quelle configuration l'application aura besoin
- La vérité est qu'il n'a pas besoin de ces informations car docker nous fournit cette abstraction
- Vous créez une image de votre application. Vous pouvez exécuter votre application partout où l'environnement d'exécution docker est installé, que ce soit sur votre machine locale, dans le centre de calculs de votre entreprise ou dans le Cloud.

Scenario 2

- Vous êtes tellement content que votre application tourne
- Quelques jours plus tard, vous vous dirigez vers votre ami et lui demande:
- Hey, je veux être sûr que mon application s'exécute sans arrêt. D'autre part, j'attends une utilisation massive ce weekend de Black Friday et je veux être sûr qu'il y a assez d'instances de l'application pour servir mes clients. A propos, de nombreuses autres applications doivent être bientôt déployées et j'aimerais être capable de manager toutes ces applications.
- Votre copain vous réponds: **Nope, ce n'est pas ce que docker fait!**
- Découragé, vous retournez dans votre bureau, faites des recherches sur internet et vous découvrez le nouvel outil sur le marché: Kubernetes! Aha!

Architecture de Kubernetes

Orchestration:

- Gère des milliers d'instances et des milliers de microservices de manière declarative

Capacités:

- Auto-scaling
- Service discovery
- Load balancing
- Self-healing
- Zero-downtime deployment

Cloud neutral:

- Plateforme standardisé sur toute infrastructure Cloud (Azure, AWS, GC)

Kubernetes

Scenario 2 - Suite

- Vous retournez voir votre ami pour lui montrer votre découverte. Entre temps vous avez installé l'outil Kubernetes, et vous lui montrez comment executer votre demande de l'autre jour.
- Vous lui dites: c'était facile!... Et vous tapez les commandes suivantes:
 - `kubectl create deployment hello-world-rest-api --image=adanlessossi/hello-world-rest-api:0.0.1.RELEASE`
 - `kubectl expose deployment hello-world-rest-api --type=LoadBalancer --port=8080`
- Votre ami vous répond, c'est pas si fantastique! J'avais fait le même déploiement avec docker à l'aide d'une seule commande!
- Vous lui répondez: attends pour voir! Maintenant, je peux augmenter les instances de mon application! Et vous tapez les commandes suivantes:
 - `kubectl scale deployment hello-world-rest-api --replicas=3`
- Automatiquement, 3 instances de mon application s'exécutent et la charge est distribuée entre les 3 instance à cause du Load Balancing.

Scenario 3

- Votre ami n'est toujours pas convaincu et vous demande: j'aimerais que l'application tourne en permanence, même si une instance crache, j'aimerais qu'elle soit remplacée immédiatement.
- Easy: vous exécutez la commande suivante pour supprimer le node sur lequel tourne une instance.
 - `kubectl delete pod hello-world-rest-api-58f234abcxx-kdkla7`
- Immédiatement après, nous voyons que l'application tourne toujours, même si nous avons supprimé une instance, une autre instance vient prendre tout de suite le relais.
- Vous dites à votre ami que Kubernetes fait beaucoup de magie!

Scenario 4

- Impressionné, votre ami vous fait une dernière demande: j'attends beaucoup de trafic durant le weekend. J'aimerais que le nombre d'instances augmente automatiquement pendant cette période, et dès Dimanche soir, que le nombre d'instances diminue. Et aussi en journée, j'attends beaucoup de trafic et la nuit très peu.
- Vous répondez: easy! Et vous tapez la commande suivante:
 - `kubectl autoscale deployment hello-world-rest-api --max=10 --cpu-percent=70`
- Woaw! Votre ami vous dit alors: J'ai encore une dernière demande!
- Je veux deployer une nouvelle version de mon application sans que mon application arrête d'exécuter. Je veux partir de 0.0.1.RELEASE à 0.0.2.RELEASE
- OK, vous lui dites que ça, c'est un peu complexe mais pas impossible. Vous éditez votre fichier de déploiement et exécutez une dernière commande:
 - `kubectl set image deployment hello-world-rest-api hello-world-rest-api= adanlessossi/hello-world-rest-api:0.0.2.RELEASE`
- L'ancienne version continuera à tourner pendant quelques secondes, puis la nouvelle version prend le dessus immédiatement.

considérations

- Créer une infrastructure Kubernetes est très difficile à installer et à gérer
- Docker for Windows est livré avec Kubernetes mais ne nous permet de créer qu'un seul cluster sur notre machine locale. Ceci pourrait suffire pour des petits deployments mais ne peut être utilisé en production.
- C'est pourquoi l'utilisation d'infrastructure existantes nous facilitera la tâche au début de notre apprentissage.
- Nous utiliserons Google Cloud pour commencer.
- Ceci peut être intimidant pour certains, mais vous verrez que dès que vous aurez fini cette séance, tout vous semblera facile.

Quelques histoires amusantes...

- Kubernetes est abrégé **K8S**, K, 8 lettres et le S de la fin
- Kubernetes se prononce **KU - BER - NET – EEZ**
- Le logo est un timonier, quelqu'un donnant la direction du bateau →
- Kubernetes sur le Cloud: AKS, Amazon EKS et GKE





Déploiement d'un service

- Nous voulons deployer un service sur Kubernetes. Que nous faut-il?
 1. Créer un compte google cloud.
 2. Créer un cluster Kubernetes sur Google Cloud
 3. Déployer notre application
 4. Vérifier notre application

Création d'un compte google cloud

- Dans votre navigateur favori, tapez l'URL suivant:
 - <https://cloud.google.com/>
- Si vous n'avez pas encore de compte google cloud, profitez pour vous inscrire et bénéficiez de 300\$ de crédit gratuit pour utiliser le cloud de google. Ce qui est intéressant, c'est que google ne vous chargera pas avant une durée d'essai d'un an.
- Acceptez les conditions et choisissez une region proche de vous (Belgique ou Suisse)
- Vous devrez également choisir le type de compte (individuel ou entreprise) et enfin donner un numéro de carte de crédit par lequel google vous chargera dans le future.

Type de compte



Account type ⓘ

Individual



Tax information

Tax status

Unregistered individual ▼



PAN (optional)

Permanent account number example: ABCDE1234A


TAN (optional)

Tax deduction and collection account number example:
DELA02603G

Détails de votre carte de crédit

- Afin de s'assurer de votre identité google va verifier votre carte de credit suivant la region du globe où vous vous trouvez

× Verify Mastercard **** 6000

 Mastercard **** 6000

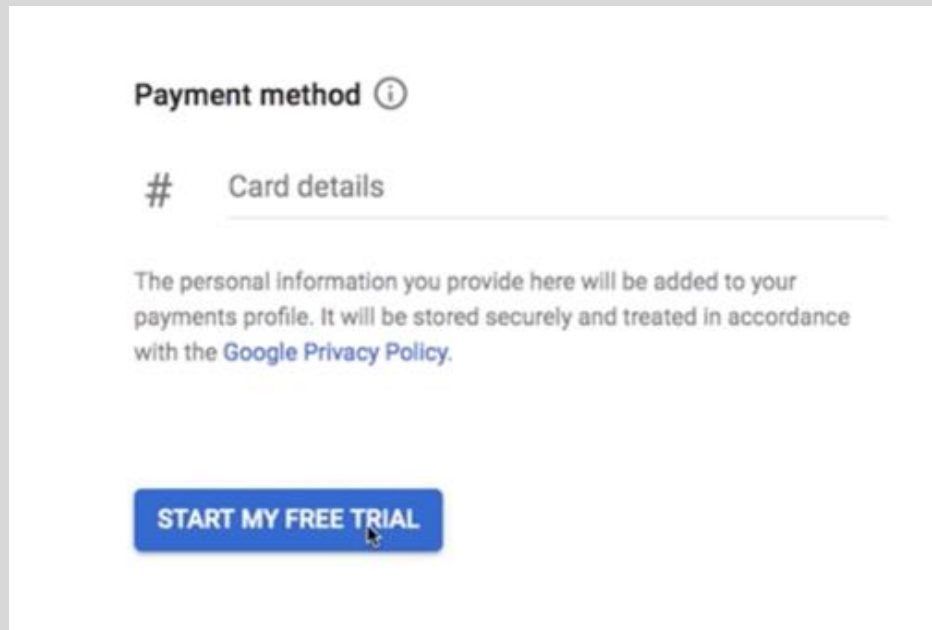
Enter the 3-digit security code on your card

Security code

CONTINUE

Démarrage de l'essai gratuit

- Après avoir inséré vos coordonnées de carte de credit et passé les verifications, cliquez sur le bouton "Start My Free Trial" pour démarrer l'essai gratuit du cloud de google.
- Félicitation!



The screenshot shows a 'Payment method' section with an information icon. Below it is a tab labeled '# Card details'. A paragraph of text explains that the provided information will be added to the user's payment profile, stored securely, and handled according to the Google Privacy Policy. At the bottom of the section is a blue button with the text 'START MY FREE TRIAL'.

Payment method ⓘ

Card details

The personal information you provide here will be added to your payments profile. It will be stored securely and treated in accordance with the [Google Privacy Policy](#).

START MY FREE TRIAL

Architecture du cluster Kubernetes

Master node:

- Gère le cluster

Worker node:

- Exécute votre application

Cluster

Gestionnaire de ressources

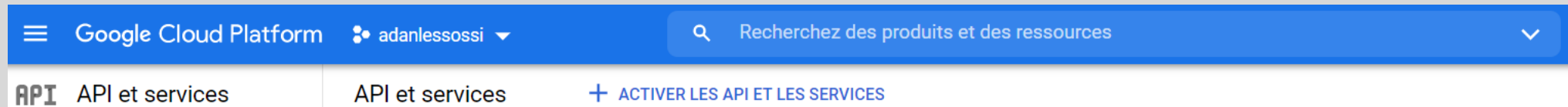
- La meilleure manière de définir Kubernetes est qu'il est un gestionnaire de ressources.
- Quelles ressources sont ici gérées? Des serveurs
- Un serveur est très souvent un serveur virtuel
 - Amazon l'appelle "EC2" (Elastic Compute Cloud)
 - Azure l'appelle "Virtual machines", et
 - Google l'appelle "Compute Engine"
 - Kubernetes l'appelle "Node"
- Donc Kubernetes peut gérer des milliers de ces nodes
- Et quand on a des milliers de choses à gérer, on introduit des gestionnaires (ici le Master node)
- Pour obtenir une haute disponibilité de l'application, notre cluster peut avoir plusieurs nodes Master

Définition d'un cluster

- Un cluster une combinaison de nodes et d'un ou plusieurs Master Nodes.
- Les nodes qui font le travail sont appelés "Worker Nodes" ou simplement nodes.
- Le node qui fait le travail de gestion est appelé "Master Node". Ce Master Node s'assure que les autres nodes soient assez motivés et les charge de travaux à effectuer.
- Un cluster est donc une grappe de nodes gérés par un Master Node.

Étapes de création de notre premier cluster

- Maintenant que nous avons tout compris, essayons de créer notre premier cluster.
- Depuis le tableau de bord, tapez “Kubernetes” dans la recherche pour sélectionner “Kubernetes Engine”



- Ceci prendra un petit moment pour activer Kubernetes Engine

Le tableau de bord de Kubernetes

- Dès l'initialisation terminée, vous verrez le tableau de bord de Kubernetes s'afficher.

The screenshot shows the Google Cloud Platform interface for Kubernetes Engine. The top navigation bar is blue with the Google Cloud Platform logo, the user name 'adanlessossi', and a search bar containing the text 'Recherchez des produits et des ressources'. Below the navigation bar, the left sidebar lists various services: Kubernetes Engine (selected), Clusters, Charges de travail, Services et entrées, Applications, Configuration, Stockage, Navigateur d'objets, and Migrer vers des conteneurs. The main content area is titled 'Clusters' and features a card for 'Kubernetes Engine Clusters Kubernetes'. This card contains a description of containers and their deployment, followed by three buttons: 'Créer un cluster' (highlighted in blue), 'Déployer un conteneur', and 'Accéder au démarrage rapide'.

Google Cloud Platform adanlessossi

Recherchez des produits et des ressources

Kubernetes Engine Clusters

Clusters

Charges de travail

Services et entrées

Applications

Configuration

Stockage

Navigateur d'objets

Migrer vers des conteneurs

Kubernetes Engine
Clusters Kubernetes

Les conteneurs créent un package d'application pour la déployer facilement en vue de l'exécuter dans son propre environnement isolé. Les conteneurs sont gérés dans des clusters qui automatisent la création et la maintenance des VM. [En savoir plus](#)

Créer un cluster Déployer un conteneur

Accéder au démarrage rapide

Création du cluster Kubernetes

- Maintenant nous pouvons cliquer sur le bouton “créer un cluster” pour créer notre premier cluster.
- Typiquement la création d'un cluster implique de savoir:
 - La mémoire à allouer
 - Le type et le nombre de node desire et leur location régionale
 - Etc
- Pour le moment acceptons les valeurs par défaut.

Paramètres de base du cluster Kubernetes

Google Cloud Platform

adanlessossi

Recherchez des produits et des ressources

← Créer un cluster Kubernetes

+ AJOUTER UN POOL DE NŒUDS

🗑 SUPPRIMER LE POOL DE NŒUDS

Paramètres de base du cluster

POOLS DE NŒUDS

default-pool

Nœuds

Sécurité

Métadonnées

CLUSTER

Automatisation

Réseau

Sécurité

Métadonnées

Fonctionnalités

Paramètres de base du cluster

Le cluster sera créé avec le nom, la version et l'emplacement que vous indiquez ici. Une fois le cluster créé, le nom et l'emplacement ne peuvent pas être modifiés.

?

Pour effectuer des tests avec un cluster abordable, sélectionnez **Mon premier cluster** dans le menu **Guides de configuration des clusters**

Nom

togojug-cluster

?

Type d'emplacement

☒ Zonal

☐ Régional

Zone

europa-west6-a

▼

?

☐ Spécifiez les emplacements de nœuds par défaut

?

Emplacements par défaut actuels : europa-west6-a

Détails du Worker Node

☰

Google Cloud Platform

adanlessossi

Recherchez des produits et des res

←

Créer un cluster Kubernetes

+ AJOUTER UN POOL DE NŒUDS

🗑 SUPPRIMER LE POOL DE NŒUDS

• Paramètres de base du cluster

POOLS DE NŒUDS

• default-pool

• Nœuds

• Sécurité

• Métadonnées

CLUSTER

• Automatisation

• Réseau

• Sécurité

• Métadonnées

• Fonctionnalités

Détails du pool de nœuds

Le cluster sera créé avec au moins un pool de nœuds. Un pool de nœuds représente un modèle des groupes de nœuds créés dans ce cluster. D'autres pools de nœuds peuvent être ajoutés ou supprimés après la création du cluster.

Nom

default-pool

Version du nœud

1.16.13-gke.401 (version maître)

Taille

Nombre de nœuds *

3

La plage d'adresses des pods limite la taille maximale du cluster. [En savoir plus](#)

☐ Activer l'autoscaling ?

☐ Spécifiez les emplacements de nœuds ?

Valeur par défaut : europe-west6-a

Automatisation

☒ Activer la mise à niveau automatique ?

Configuration du Node

Google Cloud Platform

adanlessossi

Recherchez des produits et des ressources

Créer un cluster Kubernetes

AJOUTER UN POOL DE NŒUDS

SUPPRIMER LE POOL DE NŒUDS

Paramètres de base du cluster

POOLS DE NŒUDS

default-pool

Nœuds

Sécurité

Métadonnées

CLUSTER

Automatisation

Réseau

Sécurité

Métadonnées

Fonctionnalités

Nœuds

Ces paramètres de nœud seront utilisés lorsque des nœuds seront créés à l'aide de ce pool de nœuds.

Type d'image

Container-Optimized OS (cos) (par défaut)

Configuration de la machine

Famille de machines

USAGE GÉNÉRAL

Types de machines pour les charges de travail courantes permettant d'optimiser les coûts et la flexibilité


Série

E2

Sélection de la plate-forme de processeur en fonction de la disponibilité

Type de machine

e2-medium (2 processeurs virtuels, 4 Go de mémoire)



vCPU

Un cœur partagé

Memory

4 Go

PLATE-FORME DU PROCESSEUR ET GPU

Création du cluster

- Cliquez sur le bouton “Créer” pour finaliser la creation du cluster.

| | | | |
|--|--------------------------------------|--|---|
| | | EN SAVOIR PLUS | |
| | <input type="button" value="CRÉER"/> | <input type="button" value="ANNULER"/> | Requête/Réponse REST ou ligne de commande équivalente |




- Ceci prendra au minimum 5 minutes pour la création du cluster.

Aperçu du cluster


- Finalement le cluster est créé:

Un cluster Kubernetes Engine est un groupe géré d'instances de VM pour exécuter des applications en conteneur. [En savoir plus](#)

Filtrer par étiquette ou par nom

| <input type="checkbox"/> Nom ^ | Zone | Taille du cluster | Nombre total de cœurs | Mémoire totale | Notifications | Libellés |
|--|----------------|-------------------|------------------------|----------------|---------------|--|
| <input type="checkbox"/>  togojug-cluster | europe-west6-a | 3 | 6 processeurs virtuels | 12,00 Go | | Se connecter   |

[←](#) Clusters [MODIFIER](#) [SUPPRIMER](#) [AJOUTER UN PO](#)

 togojug-cluster


[Détails](#) [Stockage](#) [Nœuds](#)

Cluster

| | | |
|--|-------------------------|---|
| Version disponible | Aucun | Modifier le canal de publication |
| Version maître | 1.16.13-gke.401 | Mise à jour disponible |
| Point de terminaison | 34.65.39.58 | Afficher le certificat du cluster |
| Certificat client | Désactivé | |
| Autorisation binaire | Désactivée | |
| Fonctionnalités alpha Kubernetes | Désactivées | |
| Taille totale | 3 | |
| Zone maître | europe-west6-a | |
| Zones de nœuds par défaut | europe-west6-a | |
| Réseau | default | |
| Sous-réseau | default | |
| VPC natif (adresse IP d'alias) | Activé | |
| Plage d'adresses du pod | 10.0.0.0/14 | |
| Nombre de pods maximum par nœud par défaut | 110 | |
| Plage d'adresses du service | 10.4.0.0/20 | |


Détail des nodes


- En cliquant sur “Nodes” on peut apercevoir le detail de nos nodes. Lors de la creation on avait alloué 4 GB de mémoire par node, maintenant nous avons 2.97GB!!! Mais qui utilise la mémoire allouée??
- La réponse est Kubernetes. Le Master réclame un peu de mémoire par node pour pouvoir le gérer!




 togojug-cluster

[Détails](#) [Stockage](#) [Nœuds](#)

Nœuds

 Filtrer les nœuds

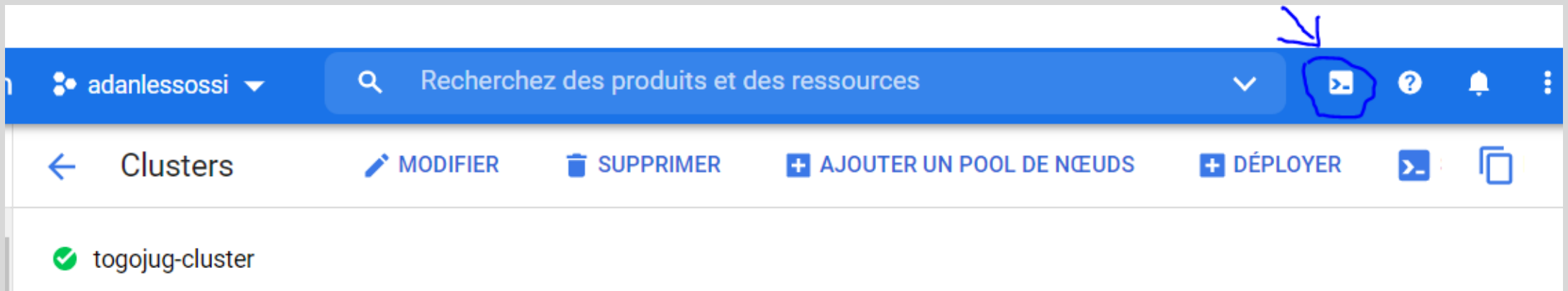
 Colonnes ▼

| Nom ^ | État | Processeur demandé | Processeur allouable | Mémoire demandée | Mémoire allouable | Stockage demandé | Stockage allouable |
|--|---|--------------------|----------------------|------------------|-------------------|------------------|--------------------|
| gke-togojug-cluster-default-pool-dc0373c6-3rmk |  Ready | 463 mCPU | 940 mCPU | 377.49 MB | 2.97 GB | 0 B | 0 B |
| gke-togojug-cluster-default-pool-dc0373c6-672b |  Ready | 483 mCPU | 940 mCPU | 387.97 MB | 2.97 GB | 0 B | 0 B |
| gke-togojug-cluster-default-pool-dc0373c6-cdnz |  Ready | 359 mCPU | 940 mCPU | 605.03 MB | 2.97 GB | 0 B | 0 B |




Connection au cluster

- Pour utiliser notre cluster il va falloir se connecter.
- Comment se connecter à un serveur distant? Nous le savons tous, en utilisant la ligne de commande. En plus il va falloir installer des utilitaires de ligne de commande.
- Google Cloud s'est arrangé pour nous rendre la vie facile. Il nous fournit ce qu'on appelle "google cloud shell"
- Assurez vous d'être dans les details du cluster et appuyez le bouton suivant:



Démarrage du shell


- Acceptez les conditions d'utilisation pour démarrer le shell.

 Cloud Shell

Cloud Shell intègre nativement l'outil gcloud du SDK Cloud, Cloud Code, un éditeur de code en ligne ainsi que d'autres utilitaires entièrement authentifiés et à jour. [En savoir plus](#)

☒ I agree that my use of any Google Cloud Platform Services is subject to my compliance with the applicable [Google Cloud Platform Terms of Service](#), [Google Cloud Privacy Policy](#), and the terms of service of [any applicable services and APIs](#).

Démarrer Cloud ShellAnnuler

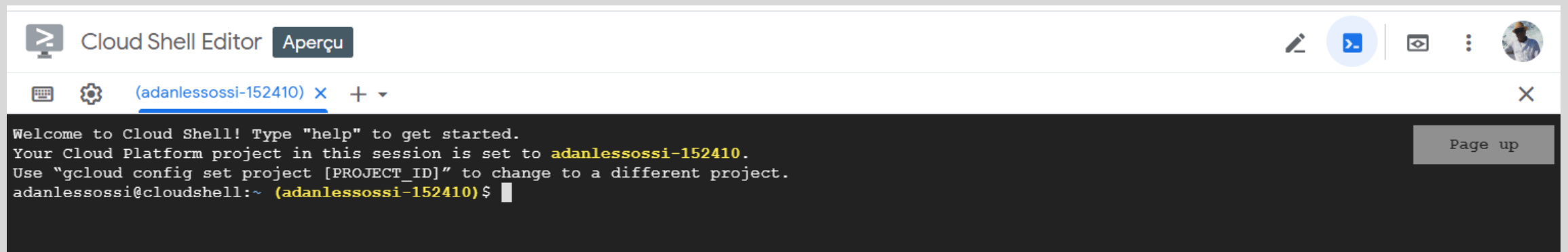
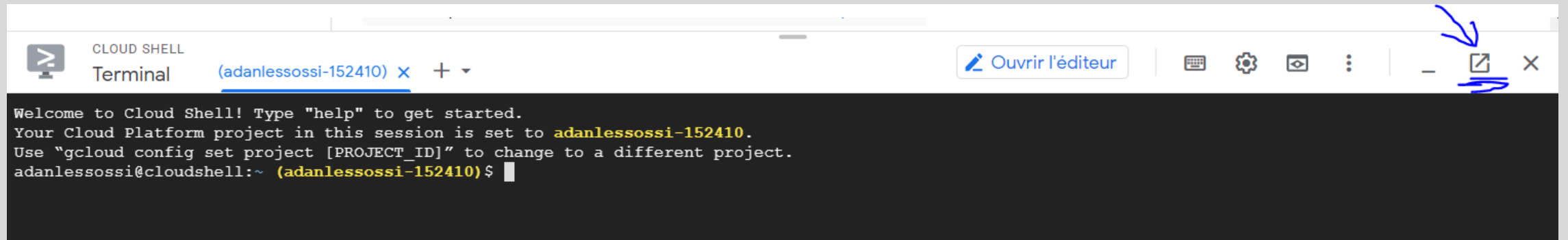
 CLOUD SHELL
Terminal (adanlessossi-152410) x + ▾

Ouvrir l'éditeur

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to adanlessossi-152410.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
adanlessossi@cloudshell:~ (adanlessossi-152410)$
```

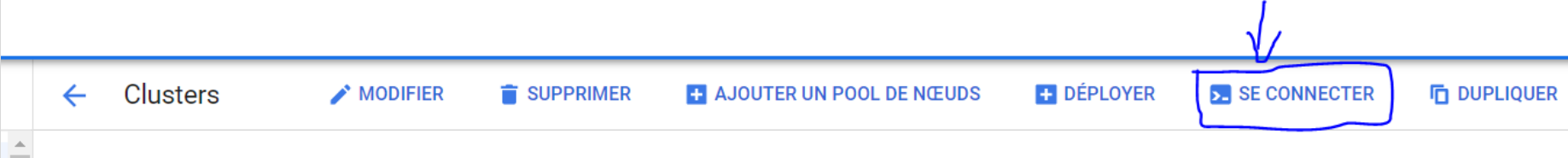
Démarrage du shell dans une nouvelle fenêtre

- Pour démarrer le shell dans une nouvelle fenêtre cliquez comme suit:



Connection au cluster

- Pour se connecter au cluster, cliquez sur bouton “Connect”



The screenshot shows the Google Cloud console interface for managing clusters. The top navigation bar includes a back arrow, the word 'Clusters', and several action buttons: 'MODIFIER' (with a pencil icon), 'SUPPRIMER' (with a trash icon), 'AJOUTER UN POOL DE NŒUDS' (with a plus icon), 'DÉPLOYER' (with a plus icon), 'SE CONNECTER' (with a terminal icon), and 'DUPLIQUER' (with a document icon). The 'SE CONNECTER' button is highlighted with a blue hand-drawn box, and a blue arrow points down to it from above.

Se connecter au cluster

Vous pouvez vous connecter à votre cluster via la ligne de commande ou le tableau de bord.

Accès à la ligne de commande

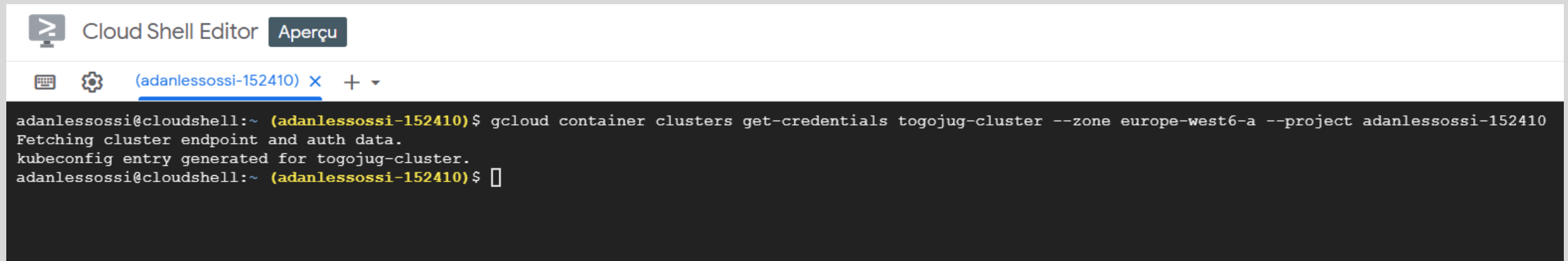
Configurez l'accès à la ligne de commande [kubectl](#) en exécutant la commande suivante :

```
$ gcloud container clusters get-credentials togojug-cluster --zone europe-west6-a --project adanlessossi-152410
```

[Exécuter dans Cloud Shell](#)

Connection au cluster - 2

- Je copie la commande et l'exécute dans cloud shell



```
adanlessossi@cloudshell:~ (adanlessossi-152410)$ gcloud container clusters get-credentials togojug-cluster --zone europe-west6-a --project adanlessossi-152410
Fetching cluster endpoint and auth data.
kubeconfig entry generated for togojug-cluster.
adanlessossi@cloudshell:~ (adanlessossi-152410)$
```

- A la fin de l'exécution de cette commande, nous nous trouvons donc dans notre cluster sur google cloud. Félicitations!

Exécution de commandes sur le cluster

- Nous voulons exécuter des commandes sur le cluster. C'est là qu'intervient la commande `kubectl`
 - `kubectl` → Contrôleur de kube
- Assurez-vous de mémoriser cette commande Kubernetes car, elle est primordiale pour gérer K8s et nous allons l'utiliser des tonnes de fois.
- `Kubectl` fonctionne avec n'importe quel cluster K8s, qu'il soit en local sur votre machine, sur un serveur de votre entreprise, sur Amazon, sur Azure ou sur Google Cloud.
- `Kubectl` peut faire beaucoup de choses avec votre cluster K8s.
- `Kubectl` est déjà installé dans Google Cloudshell

```
adanlessossi@cloudshell:~ (adanlessossi-152410)$ kubectl version
Client Version: version.Info{Major:"1", Minor:"19", GitVersion:"v1.19.3", GitCommit:"1e11e4a2108024935ecfcb2912226cedaafd99df", GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z", GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"16+", GitVersion:"v1.16.13-gke.401", GitCommit:"eb94c181eea5290e9da1238db02cfef263542f5f", GitTreeState:"clean", BuildDate:"2020-09-09T00:57:35Z", GoVersion:"go1.13.9b4", Compiler:"gc", Platform:"linux/amd64"}
adanlessossi@cloudshell:~ (adanlessossi-152410)$
```


Processus de déploiement sur le cluster

- Pour deployer votre service sur un cluster Kubernetes, il faut:
 1. Créer une image docker de votre code
 2. Pushez votre image sur un repository docker (dockerhub)
 3. Utilisez votre image pour deployer sur le cluster



Déploiement de hello-world sur le cluster

- Maintenant essayons de déployer notre application sur le cluster avec la commande:
 - `kubectl create deployment hello-world-rest-api --image=adanlessossi/hello-world-rest-api:0.0.1.RELEASE`
- Cette commande nous permet de déployer notre application sur le cluster. Mais maintenant, comment exposer l'application au monde extérieur? En exécutant la commande suivante:
 - `kubectl expose deployment hello-world-rest-api --type=LoadBalancer --port=8080`
-



Cloud Shell Editor

Aperçu



cloudshell x + ▾

```
Welcome to Cloud Shell! Type "help" to get started.
To set your Cloud Platform project in this session use "gcloud config set project [PROJECT_ID]"
adanlessossi@cloudshell:~$ kubectl create deployment hello-world-rest-api --image=adanlessossi/hello-world-rest-api:0.0.1.RELEASE
deployment.apps/hello-world-rest-api created
adanlessossi@cloudshell:~$ kubectl expose deployment hello-world-rest-api --type=LoadBalancer --port=8080
service/hello-world-rest-api exposed
adanlessossi@cloudshell:~$
```

Aperçu du service hello-world

- Cliquez sur “Services” pour voir le service installé.
- Pour ouvrir l'application dans une autre fenêtre, cliquez sur le “Point de terminaison”.

The screenshot shows the Google Cloud Platform console interface. The top navigation bar includes the Google Cloud Platform logo, the user name 'adanlessossi', and a search bar. The left sidebar lists various Kubernetes Engine resources, with 'Services et entrées' selected. The main content area displays the 'Services et entrées' page, which includes a description of services and a table of installed services. The table has columns for 'Nom', 'État', 'Type', 'Points de terminaison', 'Pods', 'Espace de noms', and 'Cluster'. The service 'hello-world-rest-api' is listed with a status of 'OK' and an external load balancer type. A blue arrow points to the checkbox next to the service name, and a blue circle highlights the IP address in the 'Points de terminaison' column.

Google Cloud Platform | adanlessossi | Recherchez des produits et des ressources

Kubernetes Engine | Services et entrées | ACTUALISER | CRÉER UN INGRESS | SUPPRIMER

Cluster | Espace de noms | RÉINITIALISER | SAVE | BÊTA

SERVICES | ENTRÉE

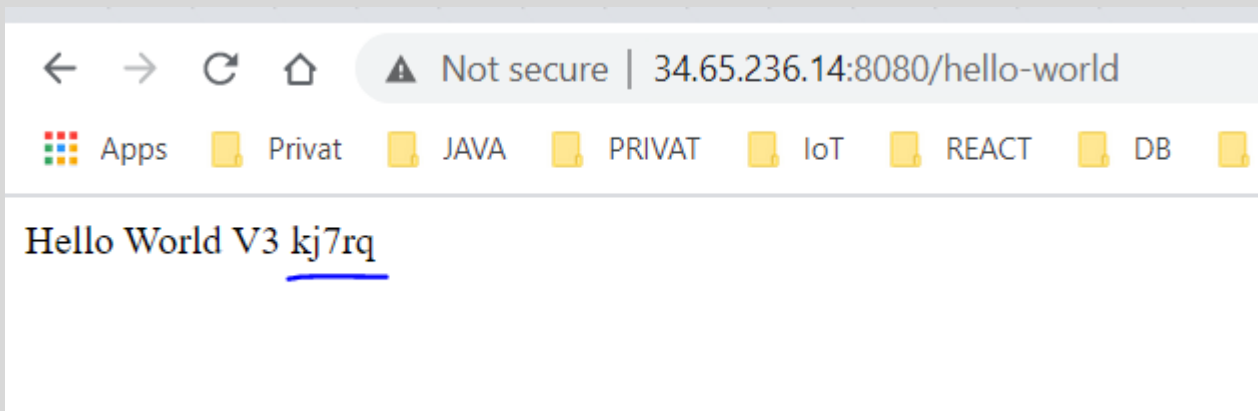
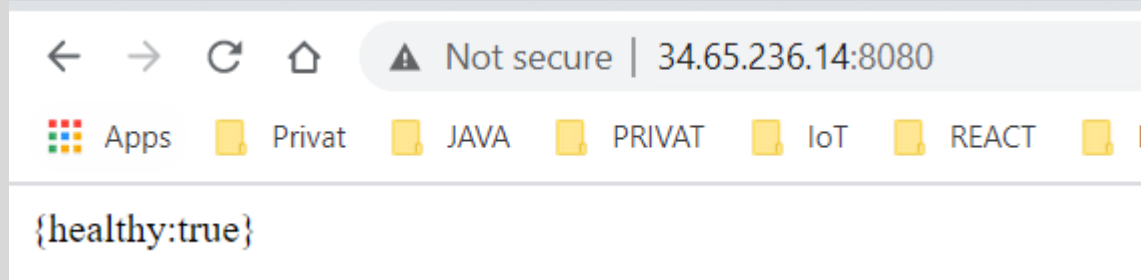
Les services sont des ensembles de pods avec un point de terminaison de réseau pouvant servir pour la détection et l'équilibrage de charge. Les entrées sont des ensembles de règles qui permettent de router le trafic HTTP(S) externe vers les services.

Est un objet système : Faux | Filtrer les services et les Ingress

| | Nom ↑ | État | Type | Points de terminaison | Pods | Espace de noms | Cluster |
|--------------------------|----------------------|------|-------------------------------|-----------------------|------|----------------|-----------------|
| <input type="checkbox"/> | hello-world-rest-api | OK | Équilibreur de charge externe | 34.65.236.14:8080 | 1/1 | default | togojug-cluster |

L'application hello-world

- Nous remarquons que l'application hello-world est healthy (sans erreur)



Prochaines étapes

- Dans les prochaines séances, nous allons:
 - Analyser les details des concepts de Kubernetes:
 - Qu'est ce qu'un Pod, un Replica Set, un déploiement
 - Approfondissement sur le Node Maître et les Worker Nodes
 - Installer des outils de déploiement sur notre machine locale (kubectl & Gcloud)
 - Comprendre ce qu'est "Infrastructure comme Code"
 - Développer 2 services pour échanger les devices et les convertir (service-échange et service-conversion)
 - Créer les fichiers d'infrastructure pour deployer ce service sur le cluster

CONCLUSION

- Ceci n'est qu'une introduction aux nouveaux pouvoirs qui sont donnés aux développeurs.
- Plus de renseignements:
- <https://kubernetes.io/>
- <https://docs.docker.com/>
- <https://hub.docker.com/>
- Les slides sont sur github
- <https://github.com/adanlessossi/kubernetes-workshop>



Mot de fin

- “***With great powers comes great responsibilities***”, Peter Parker, The Spiderman

