

# Enrichir votre application Bluemix avec les services

Dans ce chapitre, vous allez enrichir votre application nouvellement créée avec des services additionnels provenant du catalogue Bluemix.

A travers cet exercice, vous allez créer une application basée sur le stack CLEAN (CLOUDant NoSQL database, Express, Angular et Node.js).



Fork me on GitHub

Type a new todo

☐ subscribe to Bluemix

☐ deploy to Bluemix

☐ create a tutorial

☐ Select All Clear selected

# Objectif

Dans l'exercice suivant, vous allez apprendre à :

- Déployer une nouvelle application Cloud Foundry basée sur le runtime Node.js
- Créer un nouveau service de base de données Cloudnaut pour stocker vos données NoSQL
- Utiliser la ligne de commande Cloud Foundry

## Prérequis

- Avoir un [Bluemix IBM id](#), ou utiliser son compte existant.
- Installer le [Bluemix CLI](#)
- Installer un [Git client](#)
- Installer [Node.js](#)

# Etapes

1. Récupérer le code source de l'application Todo
2. Créer et associer le service Cloudant
3. Connecter la DB Cloudant avec le code de l'application
4. Exécuter l'application Todo localement
5. Pousser votre code local sur le cloud

## Etape 1 - Récupérer le code source de l'application Todo

Dans le chapitre précédent, nous avons vu les bases pour modifier et déployer une application. Concentrons nous maintenant pour concevoir une application de Todo liste. L'application a déjà été développée et est disponible dans ce dépôt Git.

Votre première tâche consiste à intégrer ce code dans l'application que vous venez de créer, en remplaçant le code existant.

1. Supprimer tous les fichiers et dossiers de votre application **sauf le fichier manifest.yml et le dossier .git.**

1. Télécharger l'application complete Todo depuis [cette archive](#) dans un répertoire temporaire.
2. Extraire les fichiers dans un répertoire temporaire.. Cela va créer un dossier *node-todo-master*.
3. Copier tous ces fichiers et dossiers vers le dossier de votre application **webapp-[your-initials]**.

Note: Assurez vous que les fichiers cachés (.gitignore, .cfignore et .bowerrc) seront aussi copiés.

## Etape 2 - Créer et associer le service Cloudant

Afin d'enregistrer nos todos, nous aurons besoin d'un stockage persistant. Pour cela, nous allons utiliser une base de données Cloudant NoSQL, base de données documents, compatible avec CouchDB.



1. Revenir à la console Bluemix, allez sur le menu **Overview** de votre application.
2. Cliquer sur **Connect New** pour ajouter un service à votre application.
3. Rechercher **Cloudant** dans le catalogue
4. Choisir le plan gratuit **Lite**
5. Donner un nom à votre service comme **webapp-cloudant-[your-initials]**
6. Cliquer sur **Create**. Bluemix va ainsi provisionner une base de données Cloudant et la connecter à votre application.
7. Choisir **Restage** quand on vous le demande.



Note: Toutes ces étapes sont réalisables en ligne de commande:

```
cf create-service cloudantNoSQLDB  
Lite webapp-cloudant-[your-initials]
```

```
cf bind-service webapp-[your-initials]  
webapp-cloudant-[your-initials]
```

```
cf restage webapp-[your-initials]
```

## Etape 3 - Connecter la DB Cloudant avec le code de l'application

Quand votre application s'exécute sur Cloud Foundry, toutes les informations des services associés à votre application sont disponible dans la variable d'environnement **VCAP\_SERVICES**.

Afin de tester cette approche, nous allons créer un fichier local d'environnement (JSON), et valider si les valeurs sont bien chargées avec notre application déployée localement.

1. Dans la console Bluemix, aller sur le dashboard de votre application.
2. Choisir **Runtime**, et **Environment Variables**
3. Copier tout le contenu de **VCAP\_SERVICES** dans le fichier local vcap-local.json de votre projet. S'assurer de copier le contenu en dessous de l'élément services.

Cela doit ressembler à ça:

```
```json
{
  "services":
  {
    "cloudantNoSQLDB": [
      {
        "credentials": {
          "username": "XXXX",
          "password": "XXXX",
          "host": "XXXXXX-bluemix.cloudant.com",
          "port": 443,
          "url": "https://....-bluemix.cloudant.com"
        },
        "name": "todo-cloudant",
        "label": "cloudantNoSQLDB",
        "plan": "Lite",
        ...
      }
    ]
  }
}
```
```

## Etape 4 - Exécuter l'application Todo localement

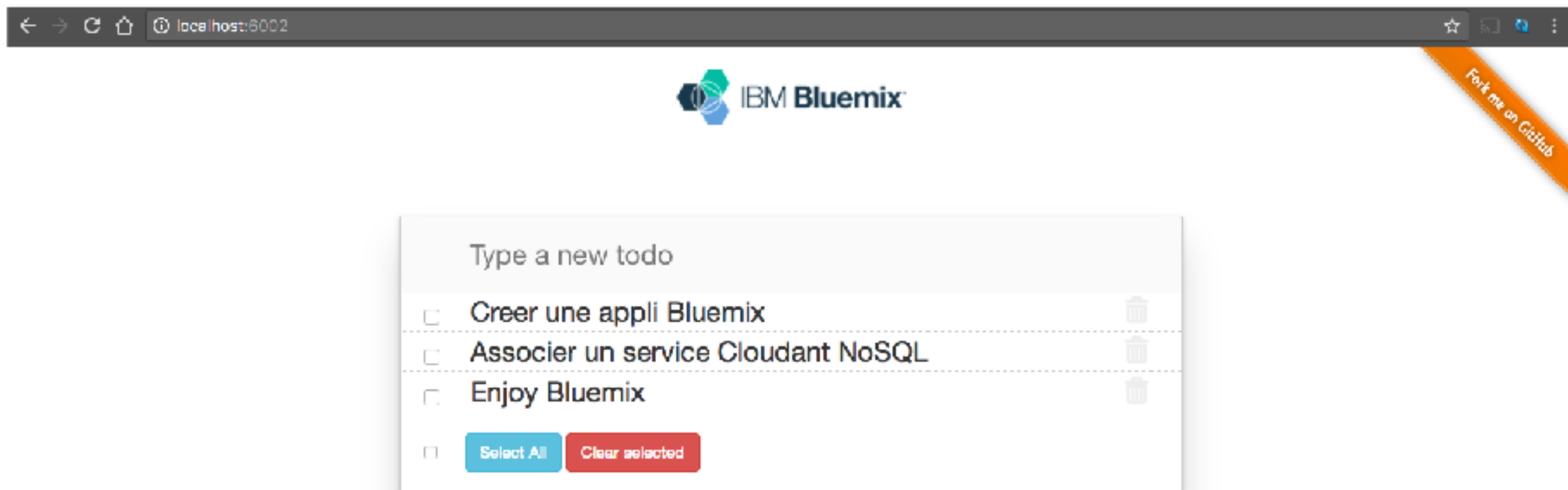
1. Installer les dépendances node.js

```
npm install
```

2. Démarrer l'application

```
npm start
```

## 1. Accéder à l'application avec votre navigateur web



## Etape 5 - Pousser votre code local sur le cloud

1. Se connecter à Bluemix en indiquant le endpoint Bluemix de l'URL avec la région où l'application a été créée.

```
bx api https://api.eu-gb.bluemix.net
```

2. S'authentifier à Bluemix

```
bx login
```

3. Pousser l'application sur Bluemix

```
bx cf push
```



- Quand la commande est terminée, accéder à l'application s'exécutant dans le cloud pour confirmer que le changement a été déployé




```
requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: webapp-jd.eu-gb.mybluemix.net
last uploaded: Wed Jun 7 14:46:39 UTC 2017
stack: unknown
buildpack: sdk-for-nodejs
```

|    | state   | since                  | cpu  | memory    | disk    | details |
|----|---------|------------------------|------|-----------|---------|---------|
| #0 | running | 2017-06-07 04:47:58 PM | 0.0% | 0 of 256M | 0 of 1G |         |



Fork me on GitHub

Type a new todo

- ☐ Créer une appli Bluemix 
- ☐ Associer un service Cloudant NoSQL 
- ☐ Enjoy Bluemix 
- ☐ Select All Clear selected

Félicitations ! Vous avez complété cet exercice.

Vous pouvez prendre connaissance du code source de l'application.

## Back-end

| File                | Description  |
|---------------------|--|
| <b>package.json</b> | Lists the node.js dependencies   |
| <b>.cfignore</b>    | List of files and directories ignored when calling <b>cf push</b> . Typically we ignore everything that can be retrieved with bower or npm. This speeds up the push process. |
| <b>manifest.yml</b> | Used by Cloud Foundry when pushing the application to define the application environment, connected services, number of instances, etc.                                      |
| <b>app.js</b>       | Web app backend entry point. It initializes the environment and imports the Todo API endpoints   |
| <b>todos.js</b>     | Todo API implementation. It declares endpoints for PUT/GET/DELETE (create/retrieve/delete) and handles the <i>in-memory</i> storage.   |

## Front-end

| File                      | Description  |
|---------------------------|--|
| <b>.bowerrc</b>           | Configuration file for the <a href="#">bower</a> web package manager to put our web dependencies under public/vendor |
| <b>bower.json</b>         | Web dependencies (bootstrap, angular)  |
| <b>index.html</b>         | Web front-end implementation. It displays the todo list and has a form to submit new todos.                          |
| <b>todo.js</b>            | Declares the Angular app   |
| <b>todo.service.js</b>    | Implements the connection between the front-end and the back-end. It has methods to create/retrieve/delete Todos     |
| <b>todo.controller.js</b> | Controls the main view, loading the current todos and adding/removing todos by delegating to the Todo service        |

# Resources

For additional resources pay close attention to the following:

- [GitHub Guides](#)
- [Get started guides for your favorite runtimes](#)

Suivre le chapitre suivant [DevOps avec Bluemix](#).

**Enjoy Bluemix !** 