Getting started with SDK for Node.js

Last Updated: 2019-11-07

Congratulations, you deployed a Hello World sample application on IBM Cloud®! To get started, follow this step-by-step guide. Or, download the sample code and explore on your own.

By following this tutorial, you'll set up a development environment, deploy an app locally on IBM Cloud®, and integrate an IBM Cloud database service in your app.

Tip: Throughout these docs, references to the Cloud Foundry CLI are now updated to the IBM Cloud CLI! The IBM Cloud CLI has the same familiar Cloud Foundry commands, but with better integration with IBM Cloud accounts and other services. Learn more about getting started with the IBM Cloud CLI in this tutorial.

Before you begin

You'll need the following accounts and tools:

- IBM Cloud account
- IBM Cloud CLI
- Git
- Node

Step 1: Clone the sample app

First, clone the Node.js *hello world* sample app GitHub repo.

```
git clone https://github.com/IBM-Cloud/get-started-node
```

Step 2: Run the app locally

Use the npm package manager to install dependencies and run your app.

1. On the command line, change the directory to where the sample app is located.

```
cd get-started-node
```

2. Install the dependencies listed in the package.json file to run the app locally.

```
npm install
```

3. Run the app.

```
npm start
```

4. View your app at the following URL: http://localhost:3000

Tip: Use nodemon for automatic restarting of application on file changes.

Step 3: Prepare the app for deployment

To deploy to IBM Cloud, it can be helpful to set up a manifest.yml file. The manifest.yml includes basic information about your app, such as the name, how much memory to allocate for each instance and the route. We've provided a sample manifest.yml file in the `get-started-node` directory.

Open the manifest.yml file, and change the `name` from `GetStartedNode` to your app name, *app_name*.

```
applications:
- name: GetStartedNode
  random-route: true
  memory: 128M
```

Tip: In this manifest.yml file, **random-route: true** generates a random route for your app to prevent your route from colliding with others. If you choose to, you can replace **random-route: true** with **host: myChosenHostName**, supplying a host name of your choice.

Step 4: Deploy the app

You can use the IBM Cloud CLI to deploy apps to IBM Cloud.

1. Log in to your IBM Cloud account, and select an API endpoint.

```
ibmcloud login
```

If you have a federated user ID, instead use the following command to log in with your single sign-on ID. See Logging in with a federated ID for more information.

```
ibmcloud login --sso
```

2. Target a Cloud Foundry org and space:

```
ibmcloud target --cf
```

If you don't have an org or a space set up, see Adding orgs and spaces.

3. From within the *get-started-node* directory, push your app to IBM Cloud.

```
ibmcloud cf push
```

Deploying your application can take a few minutes. When deployment completes, you'll see a message that your app is running. View your app at the URL listed in the output of the push command, or view both the app deployment status and the URL by running the following command:

```
ibmcloud cf apps
```

You can also go to the IBM Cloud resource list to view your app.

Tip: You can troubleshoot errors in the deployment process by using the `ibmcloud cf logs <Your-App-Name> --recent` command.

Step 5: Add a database

Next, we'll add an IBM Cloudant NoSQL database to this application and set up the application so that it can run locally and on IBM Cloud.

1. In your browser, log in to IBM Cloud and go to the Dashboard. Select **Create resource**.
2. Search for **IBM Cloudant**, and select the service.
3. For **Available authentication methods**, select **Use both legacy credentials and IAM**. You can leave the default settings for the other fields. Click **Create** to create the service.
4. In the navigation, go to **Connections**, then click **Create connection**. Select your application, and click **Connect**.
5. Using the default values, click **Connect & restage app** to connect the database to your application. Click **Restage** when prompted.

   IBM Cloud will restart your application and provide the database credentials to your application using the `VCAP_SERVICES` environment variable. This environment variable is available to the application only when it is running on IBM Cloud.

Tip: Environment variables enable you to separate deployment settings from your source code. For example, instead of hardcoding a database password, you can store it in an environment variable that you reference in your source code.

Step 6: Use the database

We're now going to update your local code to point to this database. We'll create a JSON file that will store the credentials for the services the application will use. This file will get used ONLY when the application is running locally. When running in IBM Cloud, the credentials will be read from the `VCAP_SERVICES` environment variable.

3. In the `get-started-node` directory, create a file called `vcap-local.json` with the following content:

```
{
  "services": {
    "cloudantNoSQLDB": [
      {
        "credentials": {
          "url":"CLOUDANT_DATABASE_URL"
        },
        "label": "cloudantNoSQLDB"
      }
    ]
  }
}
```

2. Find your app in the IBM Cloud resource list. On the Service Details page for your app, click **Connections** in the sidebar. Click the IBM Cloudant menu icon (...) and select **View credentials**.

3. Copy and paste just the `url` from the credentials to the `url` field of the `vcap-local.json` file, replacing `CLOUDANT_DATABASE_URL`.

4. Run your application locally.

```
npm start
```

View your local app at http://localhost:3000. Any names you enter into the app will now get added to the database.

Avoid trouble: IBM Cloud defines the PORT environment variable when your app runs on the cloud. When you run your app locally, the PORT variable is not defined, so 3000 is used as the port number. See Run your app locally for more information.

Your local app and the IBM Cloud app are sharing the database. Names you add from either app will appear in both when you refresh the browsers.

Tip: Remember, if you don't need your app live on IBM Cloud, stop the app so you don't incur any unexpected charges.

Next steps

- Tutorials
- Samples
- Architecture Center

FEEDBACK