

Tutorial

Uma vez que todas as ferramentas foram instaladas, pode-se começar a etapa de codificação.

Mas antes, conversaremos sobre o YARN.

Yarn e NPM são gestores de pacotes, que cumprem basicamente a mesma missão. O Yarn nasceu dentro do facebook.



Tutorial

Para uma comparação mais exaustiva teríamos de comparar versão a versão de NPM e Yarn. Mas basicamente fazem o mesmo, e (por enquanto) são compatíveis pois ambas usam o package.json como fonte de informação de que pacotes e versões o projeto precisa. Mas atenção: como ambas escrevem (e apagam) pacotes/programas do diretório node_modules pode acontecer que um instale/apague versões específicas que o outro instalou/removeu.

Diferenças na API:

NPM		YARN
npm init		yarn init
npm install ...		yarn add ...
npm update ...		yarn upgrade ...
npm remove ...		yarn remove ...

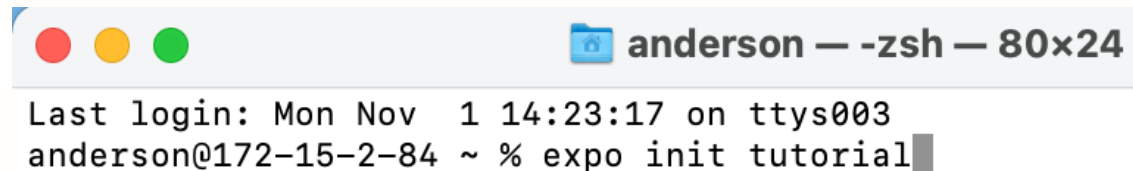
Ref: <https://pt.stackoverflow.com/questions/238580/quais-sao-diferen%C3%A7as-entre-o-npm-e-o-yarn>



Tutorial

Desenvolvimento do aplicativo:

1) Abrir o terminal e digitar **expo init tutorial**

A screenshot of a terminal window with a light gray title bar. The title bar contains three colored window control buttons (red, yellow, green) on the left and a folder icon followed by the text "anderson — -zsh — 80x24" on the right. The terminal content shows the login history "Last login: Mon Nov 1 14:23:17 on ttys003" and the current command prompt "anderson@172-15-2-84 ~ % expo init tutorial" with a black cursor at the end of the line.

```
Last login: Mon Nov 1 14:23:17 on ttys003
anderson@172-15-2-84 ~ % expo init tutorial
```



Tutorial

Desenvolvimento do aplicativo:

2) Será questionado o tipo de template a ser utilizado

```
anderson — node /usr/local/bin/expo init tutorial — 114x24
Last login: Mon Nov  1 14:23:17 on ttys003
anderson@172-15-2-84 ~ % expo init tutorial

There is a new version of expo-cli available (4.12.11).
You are currently using expo-cli 4.10.1
Install expo-cli globally using the package manager of your choice;
for example: `npm install -g expo-cli` to get the latest version

? Choose a template: > - Use arrow-keys. Return to submit.
  ----- Managed workflow -----
  > blank a minimal app as clean as an empty canvas
    blank (TypeScript) same as blank but with TypeScript configuration
    tabs (TypeScript)  several example screens and tabs using react-navigation and TypeScript
  ----- Bare workflow -----
    minimal            bare and minimal, just the essentials to get you started
```



Tutorial

Desenvolvimento do aplicativo:

2) Será questionado o tipo de template a ser utilizado

- a) blank: aplicativo mínimo em JS com apenas um rótulo no meio da tela;
- b) blank (TypeScript): semelhante a anterior, porém usando TypeScript;
- c) tabs (TypeScript): exemplo básico com várias telas com o uso de **tab** para navegação usando TypeScript; e
- d) minimal: para a instalação customizada das ferramentas do aplicativo.



Tutorial

Desenvolvimento do aplicativo:

3) Pressionando ENTER na escolha do template, será realizado a configuração do ambiente

A terminal window header with a title bar containing three colored circles (red, yellow, green) on the left and a title text "anderson — npm install ◀ node /usr/local/bin/expo init tutorial — 114x24" on the right.

anderson — npm install ◀ node /usr/local/bin/expo init tutorial — 114x24

Last login: Mon Nov 1 14:23:17 on ttys003
[anderson@172-15-2-84 ~ % expo init tutorial

There is a new version of expo-cli available (4.12.11).
You are currently using expo-cli 4.10.1
Install expo-cli globally using the package manager of your choice;
for example: `npm install -g expo-cli` to get the latest version

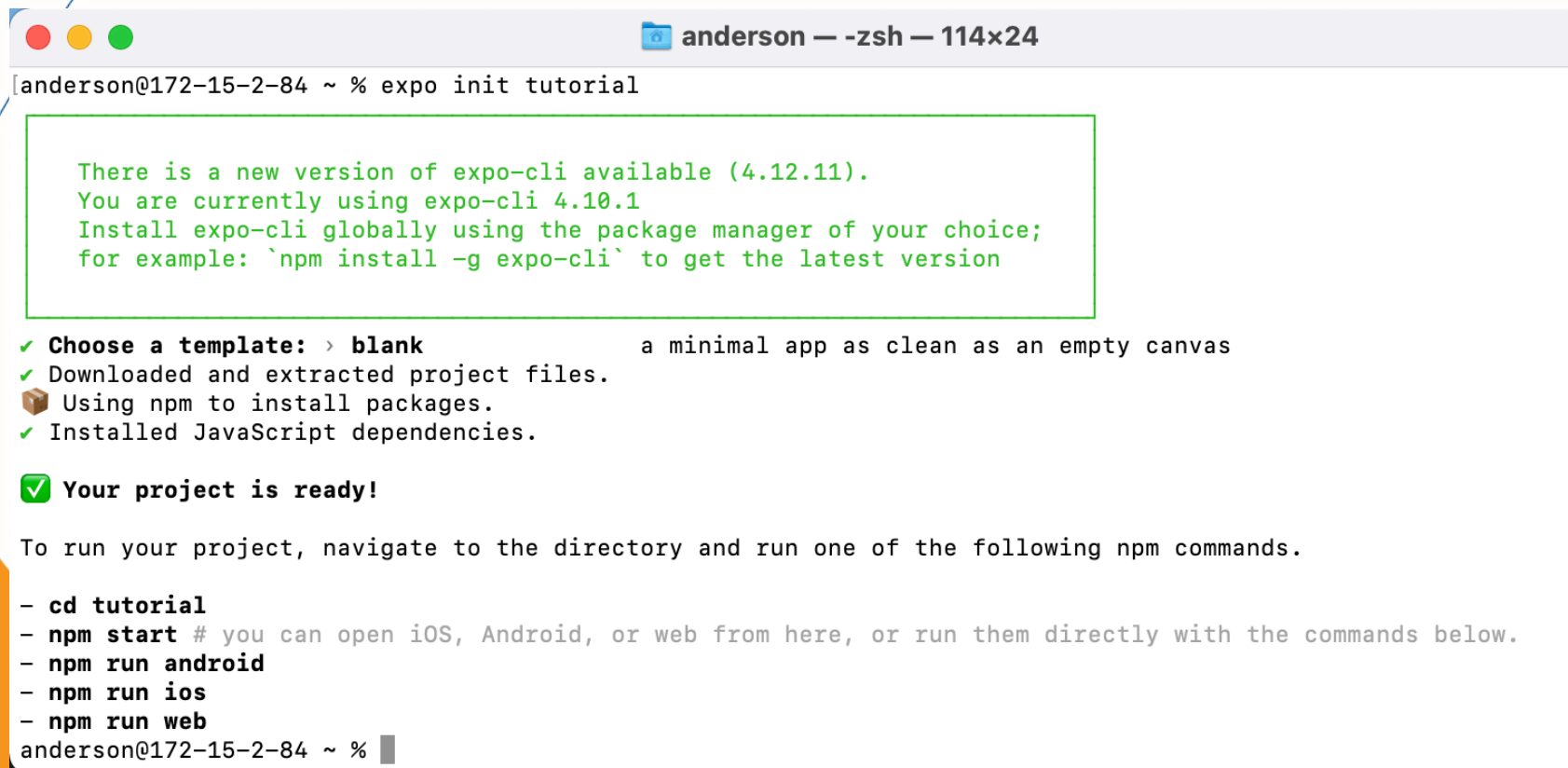
- ✓ **Choose a template:** > **blank** a minimal app as clean as an empty canvas
- ✓ Downloaded and extracted project files.
- 📦 Using npm to install packages.
- ⌚ **Installing JavaScript dependencies.**



Tutorial

Desenvolvimento do aplicativo:

4) Ao término, o usuário será informado do status da instalação e aparecerá as opções para a continuação do desenvolvimento.



```
anderson@172-15-2-84 ~ % expo init tutorial

There is a new version of expo-cli available (4.12.11).
You are currently using expo-cli 4.10.1
Install expo-cli globally using the package manager of your choice;
for example: `npm install -g expo-cli` to get the latest version

✓ Choose a template: > blank a minimal app as clean as an empty canvas
✓ Downloaded and extracted project files.
📦 Using npm to install packages.
✓ Installed JavaScript dependencies.

✓ Your project is ready!

To run your project, navigate to the directory and run one of the following npm commands.

- cd tutorial
- npm start # you can open iOS, Android, or web from here, or run them directly with the commands below.
- npm run android
- npm run ios
- npm run web
anderson@172-15-2-84 ~ %
```



Tutorial

Desenvolvimento do aplicativo:

5) Agora o passo a ser dado é mudar a pasta atual no terminal e iniciar o gerenciador para começar o desenvolvimento. É também necessário abrir o VSCode na pasta da aplicação.

- no terminal digitar **cd tutorial** e na sequência **npm start**



Tutorial

Desenvolvimento do aplicativo:

5) Agora o passo a ser dado é mudar a pasta atual no terminal e iniciar o gerenciador para começar o desenvolvimento. É também necessário abrir o VSCode na pasta da aplicação.

- no terminal digitar **cd tutorial** e na sequência **npm start**

```
tutorial — node • npm start TMPDIR=/var/folders/th/zy_4zhmd5xd1y bqdf_d6g8n
anderson@172-15-2-84 ~ % cd tutorial
anderson@172-15-2-84 tutorial % npm start
```

```
> start
> expo start
```

```
There is a new version of expo-cli available (4.12.11).
You are currently using expo-cli 4.10.1
Install expo-cli globally using the package manager of your choice;
for example: `npm install -g expo-cli` to get the latest version
```

```
Starting project at /Users/anderson/tutorial
Developer tools running on http://localhost:19002
Opening developer tools in the browser...
Starting Metro Bundler
```



Tutorial

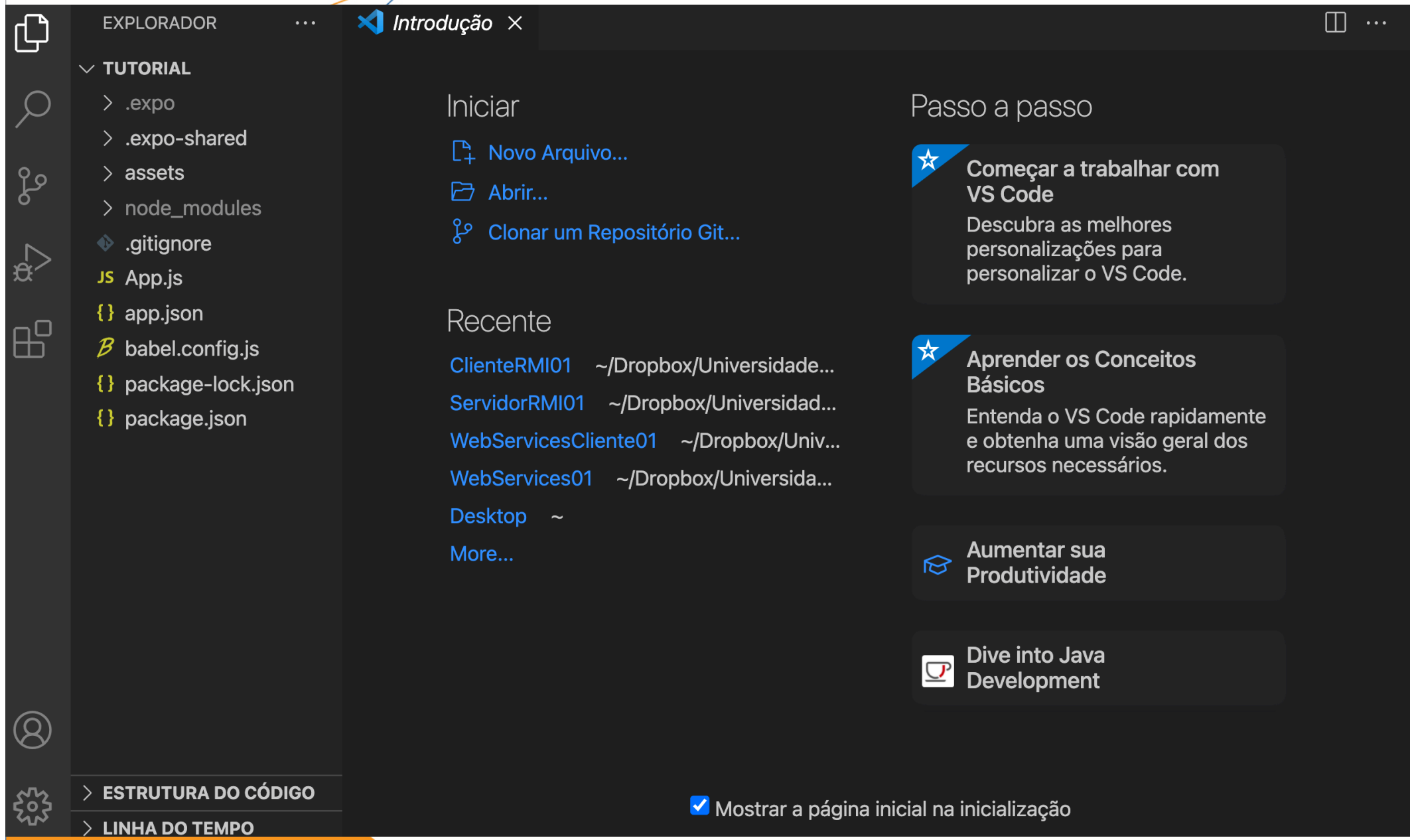
Desenvolvimento do aplicativo:

5) Agora o passo a ser dado é mudar a pasta atual no terminal e iniciar o gerenciador para começar o desenvolvimento. É também necessário abrir o VSCode na pasta da aplicação.

- no terminal digitar **cd tutorial** e na sequência **npm start**



Tutorial



EXPLORADOR

Introdução

✓ TUTORIAL

- > .expo
- > .expo-shared
- > assets
- > node_modules
- ◆ .gitignore
- JS App.js
- { } app.json
- B babel.config.js
- { } package-lock.json
- { } package.json

Iniciar

- 📄 Novo Arquivo...
- 📁 Abrir...
- 🔗 Clonar um Repositório Git...

Recente

- ClienteRMI01 ~/Dropbox/Universidade...
- ServidorRMI01 ~/Dropbox/Universidad...
- WebServicesCliente01 ~/Dropbox/Univ...
- WebServices01 ~/Dropbox/Universida...
- Desktop ~
- More...

Passo a passo

- ★ Começar a trabalhar com VS Code
Descubra as melhores personalizações para personalizar o VS Code.
- ★ Aprender os Conceitos Básicos
Entenda o VS Code rapidamente e obtenha uma visão geral dos recursos necessários.
- 🎓 Aumentar sua Produtividade
- 📖 Dive into Java Development

Mostrar a página inicial na inicialização

Tutorial

Desenvolvimento do aplicativo:

A vantagem no desenvolvimento do aplicativo através do **react-native** consiste no desenvolvimento **nativo** multiplataforma de forma simultânea. Assim são ofertados três ambientes para a execução: Android, iOS e Web.

Para o ambiente Android será acionar o **qemu**, i.e. o emulador Android presente no Android Studio onde o aplicativo será executado.

Para o ambiente iOS será acionado o **simulador iOS** e para a web será criada uma nova aba no navegador padrão do computador.

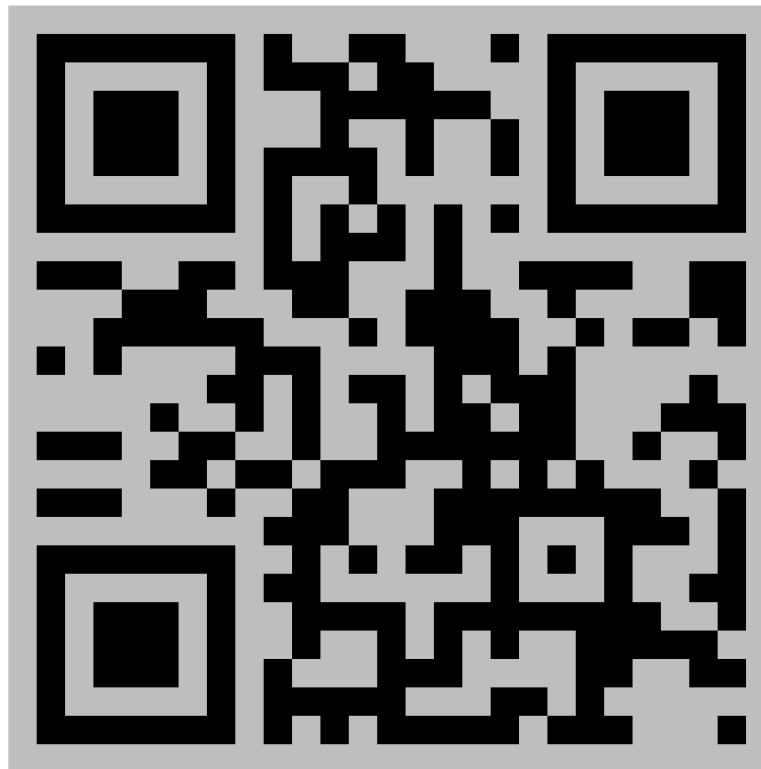
Adicionalmente o usuário poderá utilizar o próprio telefone para executar o aplicativo, através do app **expo** presente nas lojas de aplicativos. Para isto, leia o qr-code presente no terminal.



Tutorial

```
tutorial — node < npm start TMPDIR=/var/folders/th/zy_4zhmd5xd1ybqdf_d6g8mr
```

```
Developer tools running on http://localhost:19002  
Opening developer tools in the browser...  
Starting Metro Bundler
```

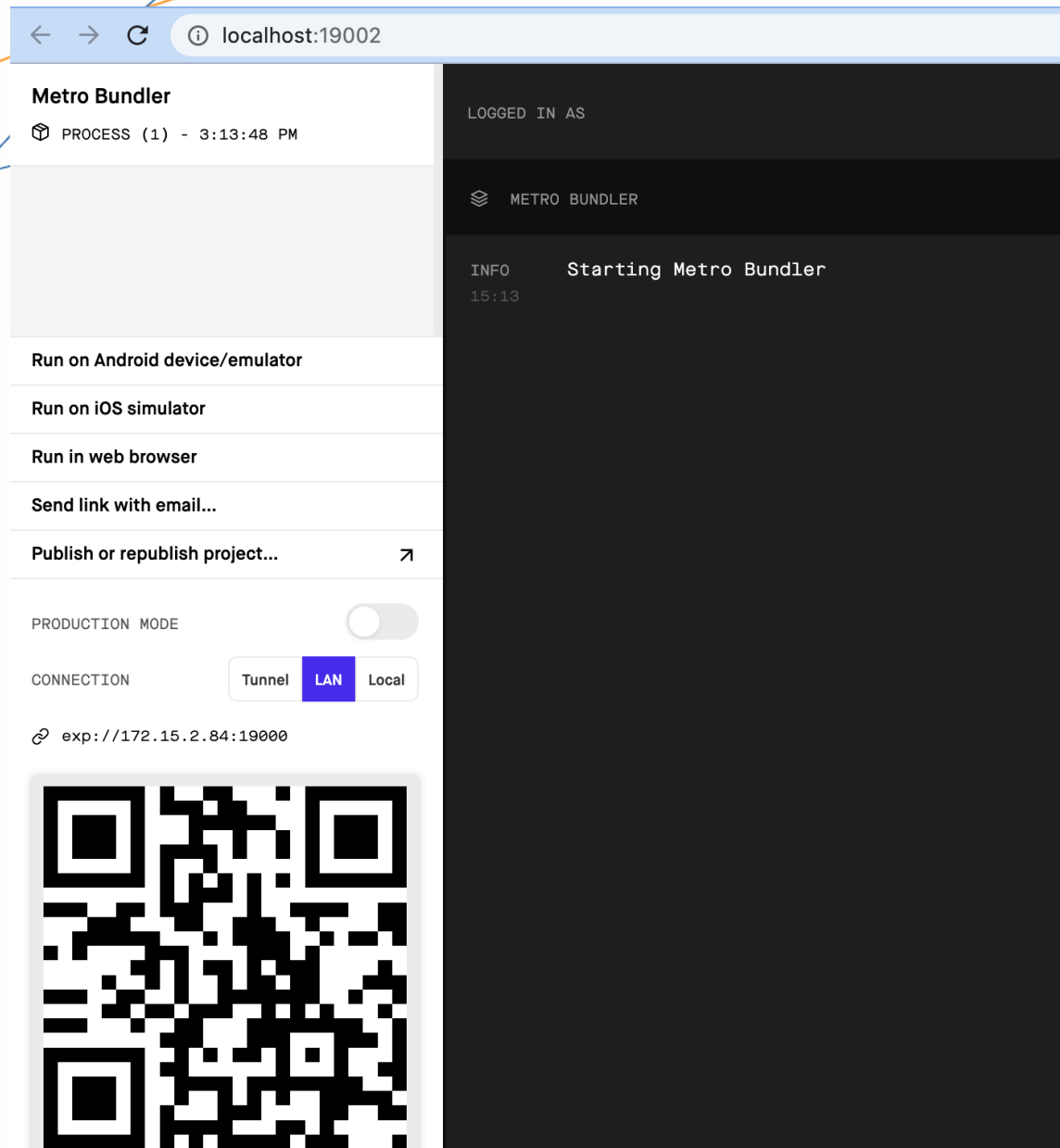


```
> Waiting on exp://172.15.2.84:19000  
> Scan the QR code above with Expo Go (Android) or the Camera app (iOS)
```



... ou na aba do navegador que foi aberta ao executar **npm start**

Tutorial



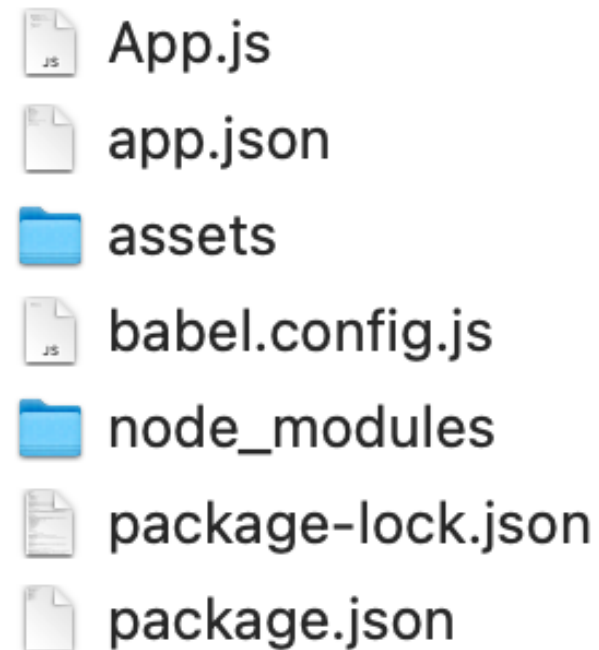
The screenshot shows a web browser window with the address bar displaying `localhost:19002`. The page is titled "Metro Bundler" and shows a process running: `PROCESS (1) - 3:13:48 PM`. The left sidebar contains several options: "Run on Android device/emulator", "Run on iOS simulator", "Run in web browser", "Send link with email...", and "Publish or republish project...". Below these options, there is a "PRODUCTION MODE" toggle switch and a "CONNECTION" section with buttons for "Tunnel", "LAN" (selected), and "Local". At the bottom of the sidebar, there is a QR code and a link: `exp://172.15.2.84:19000`. The main content area on the right shows a dark theme with the text "LOGGED IN AS" and "METRO BUNDLER". Below this, there is an "INFO" section with the text "Starting Metro Bundler" and a timestamp "15:13".



Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json
- 7) package.json



Tutorial

Estrutura de pastas criadas:

- 1) **App.js**
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json
- 7) package.json

Corresponde a aplicação principal, a partir da qual serão acionados os demais comandos e componentes que comporão a aplicação.

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) **app.json**
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json
- 7) package.json

É o manifesto do projeto. Neste arquivo pode ser encontrado, dentre outras informações detalhes de versionamento da aplicação, ícone dentre outros. Interessante notar que em outras plataformas, como o Android Studio, o arquivo usa o formato XML ao invés do formato JSON bem como possui o nome explícito de Manifesto.

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json**
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6_ package-lock.json
- 7) package.json

```
{
  "expo": {
    "name": "tutorial",
    "slug": "tutorial",
    "version": "1.0.0",
    "orientation": "portrait",
    "icon": "./assets/icon.png",
    "splash": {
      "image": "./assets/splash.png",
      "resizeMode": "contain",
      "backgroundColor": "#ffffff"
    },
    "updates": {
      "fallbackToCacheTimeout": 0
    },
    "assetBundlePatterns": [
      "**/*"
    ],
    "ios": {
      "supportsTablet": true
    },
    "android": {
      "adaptiveIcon": {
        "foregroundImage": "./assets/adaptive-icon.png",
        "backgroundColor": "#FFFFFF"
      }
    },
    "web": {
      "favicon": "./assets/favicon.png"
    }
  }
}
```

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets**
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json
- 7) package.json

Corresponde a uma pasta onde são salvos todos os **ativos** que compõem o projeto do aplicativo, como imagens.

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js**
- 5) node_modules
- 6) package-lock.json
- 7) package.json

Babel é uma ferramenta utilizada para o desenvolvimento front-end, tendo sido disponibilizada para uso em 2015. A tecnologia, chamada de transpiler (transpilador, tradutor), tem a missão de tornar o código JavaScript mais atual em um código correspondente em uma versão mais antiga.

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js**
- 5) node_modules
- 6) package-lock.json
- 7) package.json

A versão atual possui o arquivo de configurações babel.config.js, que traz complementos para o arquivo anterior chamado .babelrc. A grande diferença está na maneira como as coisas são declaradas seguindo o formato JSON, ao contrário das versões anteriores.

Ref: <https://imasters.com.br/front-end/o-que-ha-de-novo-no-babel-7>

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules**
- 6) package-lock.json
- 7) package.json

Corresponde aos arquivos de configuração do **nodejs**.

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json**
- 7) package.json

É gerado automaticamente para todas as operações em que o npm modifica a árvore node_modules ou package.json. Ele descreve a árvore exata que foi gerada, de modo que instalações subsequentes sejam capazes de gerar árvores idênticas, independentemente das atualizações de dependência intermediárias.

Tutorial

Estrutura de pastas criadas:

- 1) App.js
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6) package-lock.json
- 7) **package.json**

É um elemento-chave em muitas aplicações do ecossistema Node. ... Se você trabalha com JavaScript, ou se você já interagiu com um projeto JavaScript antes, Node.js ou front-end, você certamente já se deparou com um arquivo package.

Estrutura de pastas criadas:

- 1) **App.js**
- 2) app.json
- 3) assets
- 4) babel.config.js
- 5) node_modules
- 6_ package-lock.json
- 7) package.json

```
import { StatusBar } from 'expo-status-bar';
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default function App() {
  return (
    <View style={styles.container}>
      <Text>Open up App.js to start working on your app!
    </Text>
    <StatusBar style="auto" />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

Tutorial

```
import { StatusBar } from 'expo-status-bar';
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default function App() {
  return (
    <View style={styles.container}>
      <Text>Open up App.js to start working on your app!</Text>
      <StatusBar style="auto" />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

Importação da biblioteca da linguagem.

Tutorial

```
import { StatusBar } from 'expo-status-bar';
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default function App() {
  return (
    <View style={styles.container}>
      <Text>Open up App.js to start working on your app!</Text>
      <StatusBar style="auto" />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

Importação dos componentes utilizados no código.

Tutorial

```
import { StatusBar } from 'expo-status-bar';  
import React from 'react';  
import { StyleSheet, Text, View } from 'react-native';
```

```
export default function App() {  
  return (  
    <View style={styles.container}>  
      <Text>Open up App.js to start working on your app!</Text>  
      <StatusBar style="auto" />  
    </View>  
  );  
}
```

```
const styles = StyleSheet.create({  
  container: {  
    flex: 1,  
    backgroundColor: '#fff',  
    alignItems: 'center',  
    justifyContent: 'center',  
  },  
});
```

Importação dos componentes utilizados no código.

Tutorial

```
import { StatusBar } from 'expo-status-bar';  
import React from 'react';  
import { StyleSheet, Text, View } from 'react-native';
```

```
export default function App() {  
  return (  
    <View style={styles.container}>  
      <Text>Open up App.js to start working on your app!</Text>  
      <StatusBar style="auto" />  
    </View>  
  );  
}  
  
const styles = StyleSheet.create({  
  container: {  
    flex: 1,  
    backgroundColor: '#fff',  
    alignItems: 'center',  
    justifyContent: 'center',  
  },  
});
```

Corresponde a parte da aplicação onde será implementada a interface com o usuário e as suas funcionalidades do frontend.