Universidad Pontificia Bolivariana

Consumir recursos expuestos con Node+Express

Jefferson A. Peña Torres
jefferson.amado.pena@correounivalle.edu.co

Consumir recursos expuestos con Node+Express

Unidad 4

1.Formato JSON

2.Consumiendo API con Axios

3.Presentando datos React

La Unidad 4 de este ciclo une de manera coherente el conocimiento adquirido en las anteriores, la aplicación frontend SPA se conecta a través de HTTP a la aplicación backend para obtener recursos.

En esta unidad

- Finalizar el desarrollo de la aplicación web
- Entender el concepto de diagramas de despliegue y arquitecturas en la nube
- Subir aplicación web a un repositorio de git en la nube
- Desplegar aplicación web en una instancia de servicio en la nube con AWS
- Aplicar flujos de CI/CD para la aplicación web

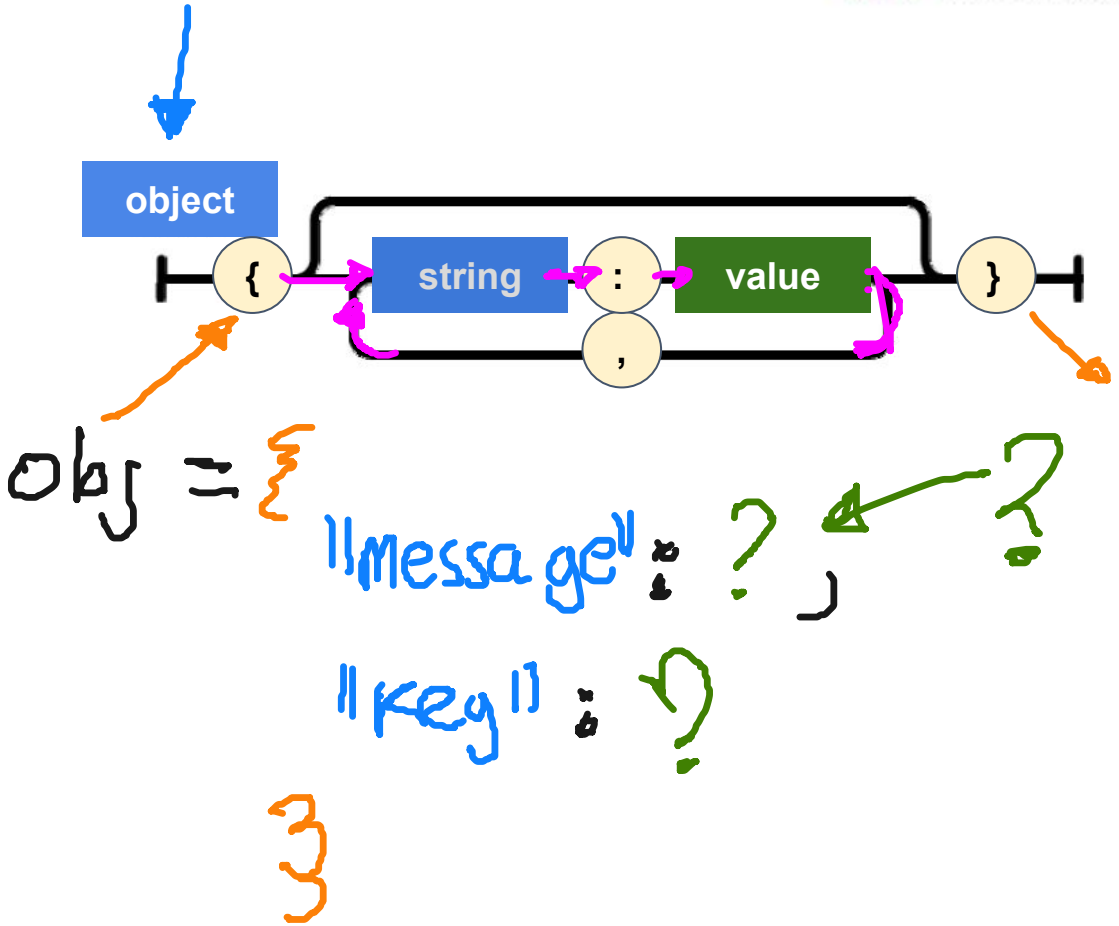
Formato JSON

Para que se logre la comunicación entre el backend y frontend es necesario un envío y recepción de información. En este caso información que debe seguir una forma o estructura que ambos entiendan. XML y JSON son de facto los formatos que permiten la transmisión en aplicaciones web modernas.

Formato JSON

Para que se logre la comunicación entre el backend y frontend es necesario un envío y recepción de información. En este caso información que debe seguir una forma o estructura que ambos entiendan. XML y JSON son de facto los formatos que permiten la transmisión en aplicaciones web modernas.

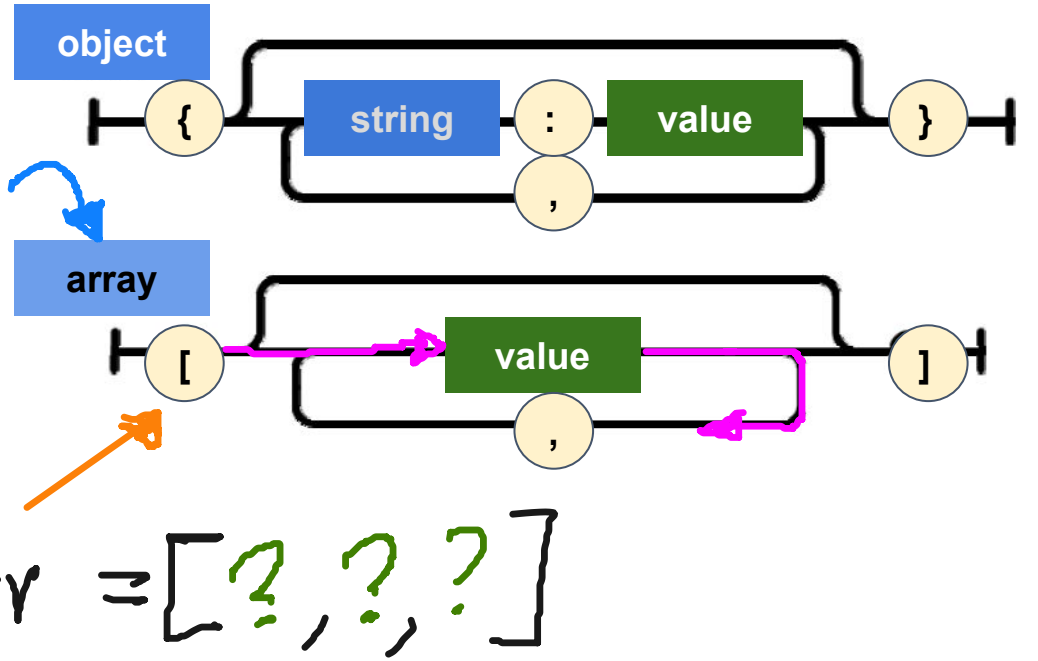
El formato o notación JSON, es ligero y tiene una sintaxis que simple que sigue un esquema que la mayoría de los lenguajes en sus versiones incorpora sin incluir plugin o intérpretes extra.



Formato JSON

Para que se logre la comunicación entre el backend y frontend es necesario un envío y recepción de información. En este caso información que debe seguir una forma o estructura que ambos entiendan. XML y JSON son de facto los formatos que permiten la transmisión en aplicaciones web modernas.

El formato o notación JSON, es ligero y tiene una sintaxis que simple que sigue un esquema que la mayoría de los lenguajes en sus versiones incorpora sin incluir plugin o intérpretes extra.

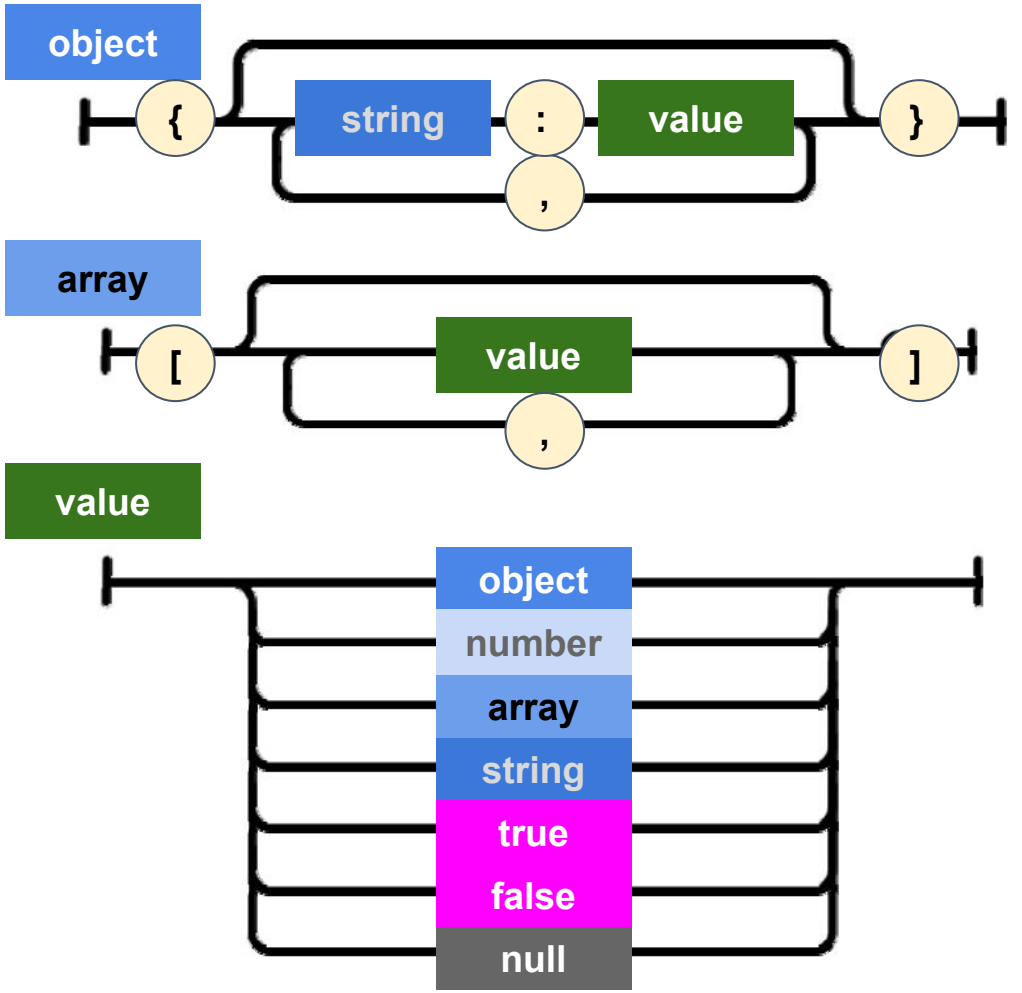


Formato JSON

Para que se logre la comunicación entre el backend y frontend es necesario un envío y recepción de información. En este caso información que debe seguir una forma o estructura que ambos entiendan. XML y JSON son de facto los formatos que permiten la transmisión en aplicaciones web modernas.

El formato o notación JSON, es ligero y tiene una sintaxis que simple que sigue un esquema que la mayoría de los lenguajes en sus versiones incorpora sin incluir plugin o intérpretes extra.

```
ot = {
    "message": [1,2]
}
```



Consumiendo API con Axios

Como lo vimos en la anterior Unidad, existen varias bibliotecas o módulos para hacer peticiones HTTP: Axios, request, superagent, request-promise, jquery y la lista continua.

	Estrellas	Forks	Creación
axios	64614	5437	Aug 19, 2014
request	23342	2723	Jan 23, 2011
superagent	14511	1210	Apr 13, 2011
request-promise	4216	277	Oct 4, 2013
jquery	52194	18494	Apr 3, 2009

Consumiendo API con Axios

1
2
3

Como lo vimos en la anterior Unidad, existen varias bibliotecas o módulos para hacer peticiones HTTP: Axios, request, superagent, request-promise, jquery y la lista continua.

Cliente HTTP basado en promesas para el navegador.

npm i axios

```
const axios = require('axios');

// Make a request for a user with a given ID
axios.get('/user/890718')
  .then(function (response) {
    // handle success
    console.log(response);
  })
  .catch(function (error) {
    // handle error
    console.log(error);
  })
  .then(function () {
    // always executed
  });
```

Consumiendo API con Axios

Como lo vimos en la anterior Unidad, existen varias bibliotecas o módulos para hacer peticiones HTTP: Axios, request, superagent, request-promise, jquery y la lista continua.

Cliente HTTP basado en promesas para el navegador.

```
const axios = require('axios');

// Make a request for a user with a given ID
axios.get('/user/90718')
  .then(function (response) {
    // handle success
    console.log(response);
  })
  .catch(function (error) {
    // handle error
    console.log(error);
  })
  .then(function () {
    // always executed
  });
```

GET
POST
PUT
DELETE

Consumiendo API con Axios

1
2
3

Como lo vimos en la anterior Unidad, existen varias bibliotecas o módulos para hacer peticiones HTTP: Axios, request, superagent, request-promise, jquery y la lista continua.

Cliente HTTP basado en promesas para el navegador.

```
const axios = require('axios');

// Make a POST requests
axios.post('/users', {
    firstName: 'Jefferson',
    lastName: 'Peña'
})
  .then(function (response) {
    console.log(response);
  })
  .catch(function (error) {
    console.log(error);
  });
```

Consumiendo API con Axios

Como lo vimos en la anterior Unidad, existen varias bibliotecas o módulos para hacer peticiones HTTP: Axios, request, superagent, request-promise, jquery y la lista continua.

Cliente HTTP basado en promesas para el navegador.

```
const axios = require('axios');

// Make a POST requests
axios.post('/users', {
    firstName: 'Jefferson',
    lastName: 'Peña'
})
.then(function (response) {
    console.log(response);
})
.catch(function (error) {
    console.log(error);
});
```

Presentando datos con React


```
import React from 'react';
import axios from 'axios';

export default class UserList extends React.Component {
  state = {
    users: []
  }
  componentDidMount() {
    axios.get(`https://localost:8000/users`)
      .then(res => {
        const users = res.data;
        this.setState({ users });
      })
  }
  render() {
    return (
      <ul>
        { this.state.users.map(u =>  <li>{u.name}</li>)}
      </ul>
    )
  }
}
```

Consumir recursos expuestos con Node+Express

Resumen

1.Formato JSON

2.Consumiendo API con Axios

3.Presentando datos React

El formato o notación **JSON**, es ligero y tiene una sintaxis que simple que sigue un esquema que la mayoría de los lenguajes en sus versiones incorpora sin incluir plugin o intérpretes extra. Este se utiliza en el intercambio de información entre Backend, con **express** u otro y en el Frontend con un cliente HTTP como lo es **Axios**.

Los datos son recibidos en el envío de solicitudes desde el frontend y de respuestas desde el backend.

www.upb.edu.co/es/mision-tic

#MisiónTICSomosTodos