

OBJETO MAP

Presentado por:
Juan Camilo Bernal Gordillo
Cristian Andres Muñoz Montenegro

MAP

🔑 Imagina que llegas a tu casa y necesitas encontrar rápidamente tus llaves. ¿Cómo las buscas?

💼 En programación, un Map es como un sistema organizado para encontrar lo que necesitas sin perder tiempo. Al igual que un llavero que organiza tus llaves, el Map organiza la información.



```
"container"
  class="row">
    class="col-md-6 col-lg-8">
      nav id="nav" role="navigation">
        ul>
          li><a href="index.html">Home</a>
          li><a href="home-events.html">Events</a>
          li><a href="multi-col-menu.html">Multi Col</a>
          li class="has-children">
            ul>
              li><a href="#">Tall Buttons</a>
              li><a href="#">Image Logo</a>
              li class="active">
                ul>
                  li><a href="#">Variable Width</a>
                ul>
              li><a href="#">Has Children</a>
                ul>
              li><a href="#">Image Logo</a>
            ul>
          li><a href="#">Image Logo</a>
        ul>
      nav>
    container>
```

¿Qué es Map?



- Un Map es una estructura de datos que almacena pares clave-valor.
- Como un contacto de teléfono, para saber su número basta con buscar su nombre.
- Permite acceder rápidamente a la información usando la clave.



```
17 string sInput;
18 int iLength, iN;
19 double dblTemp;
20 bool again = true;
21 while (again) {
22     iN = -1;
23     again = false;
24     getline(cin, sInput);
25     system("cls");
26     stringstream(sInput) >> dblTemp;
27     iLength = sInput.length();
28     if (iLength < 4) {
29         again = true;
30         continue;
31     } else if (sInput[iLength - 3] != '.') {
32         again = true;
33         continue;
34     } while (++iN < iLength) {
35         if (isdigit(sInput[iN])) {
36             again = true;
37             iLength - 3) ) {
```

Tipo de Map	Características	Cuando usarlo
⚡ HashMap	No garantiza orden, rápido, permite null en claves y valores	Cuando la velocidad importa más que el orden
🌳 TreeMap	Ordena por claves (orden natural o personalizado)	Cuando necesitas mantener los datos ordenados
🔗 LinkedHashMap	Conserva el orden de inserción	Cuando el orden de ingreso debe mantenerse
🔒 Hashtable ➡	Similar a HashMap, pero síncrono y no permite null	En entornos multihilo (aunque hoy se prefiere ConcurrentHashMap)

Implementaciones del Map en Java

```
    resp_iter = self.state.GetIterator()
    statuses = {}
    for data in resp_iter:
        status = Status(
            status_id=data.id,
            name=data.name,
            ...
        )
        statuses[status.name] = status
```

OPERACIONES BÁSICAS

- Put(clave, valor): insertar un par. 
- Get(clave): obtener el valor asociado a una clave. 
- Remove(clave): eliminar un par por su clave. 
- ContainsKey(clave): verificar si existe una clave. 
- KeySet(): obtener todas las claves. 
- Values(): obtener todos los valores. 



```
17 string sInput;
18 int iLength, iN;
19 double dblTemp;
20 bool again = true;
21 while (again) {
22     iN = -1;
23     again = false;
24     getline(cin, sInput);
25     system("cls");
26     stringstream(sInput) >> dblTemp;
27     iLength = sInput.length();
28     if (iLength < 4) {
29         again = true;
30         continue;
31     } else if (sInput[iLength - 3] != '.') {
32         again = true;
33         continue;
34     } while (++iN < iLength) {
35         if (isdigit(sInput[iN])) {
36             again = true;
37             iLength - 3) ) {
```

EJEMPLO EN JAVA



```
Hashmap.java x
1 import java.util.HashMap;
2 import java.util.Map;
3
4 public class Hashmap { new *
5     public static void main(String[] args) { new *
6         // Crear el mapa
7         Map<String, String> capitales = new HashMap<>();
8
9         // Colocar países y sus capitales
10        capitales.put("Colombia", "Bogotá");
11        capitales.put("Francia", "París");
12        capitales.put("Japón", "Tokio");
13        capitales.put("Brasil", "Brasilia");
14
15        // Obtener y mostrar la capital de Colombia
16        System.out.println("La capital de Colombia es: " + capitales.get("Colombia"));
17    }
18 }
```



EJECUCIÓN POR CONSOLA



```
Run Hashmap x
Run Stop Refresh Help
C:\Users\USUARIO\.jdks\corretto-17.0.14\bin\java.exe
La capital de Colombia es: Bogotá
Process finished with exit code 0
```



¿Por qué usar un Map?

- 1. Acceso rápido: Recuperas los valores de manera eficiente. ⏳
- 2. Organización: Ayuda a organizar los datos de manera lógica. 📁
- 3. Flexibilidad: Usa diferentes tipos de Map según lo que necesites. 🔧



Ejemplos Cotidianos de Uso de Map



Diccionario

- Clave: palabra (ej. gato)
- Valor: definición (animal doméstico felino)

Sistemas de Usuarios

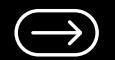
- Clave: nombre de usuario (ej. juan23)
- Valor: objeto con información del perfil

Contador de Votos

- Clave: opción (ej. "A", "B", "C")
- Valor: número de votos

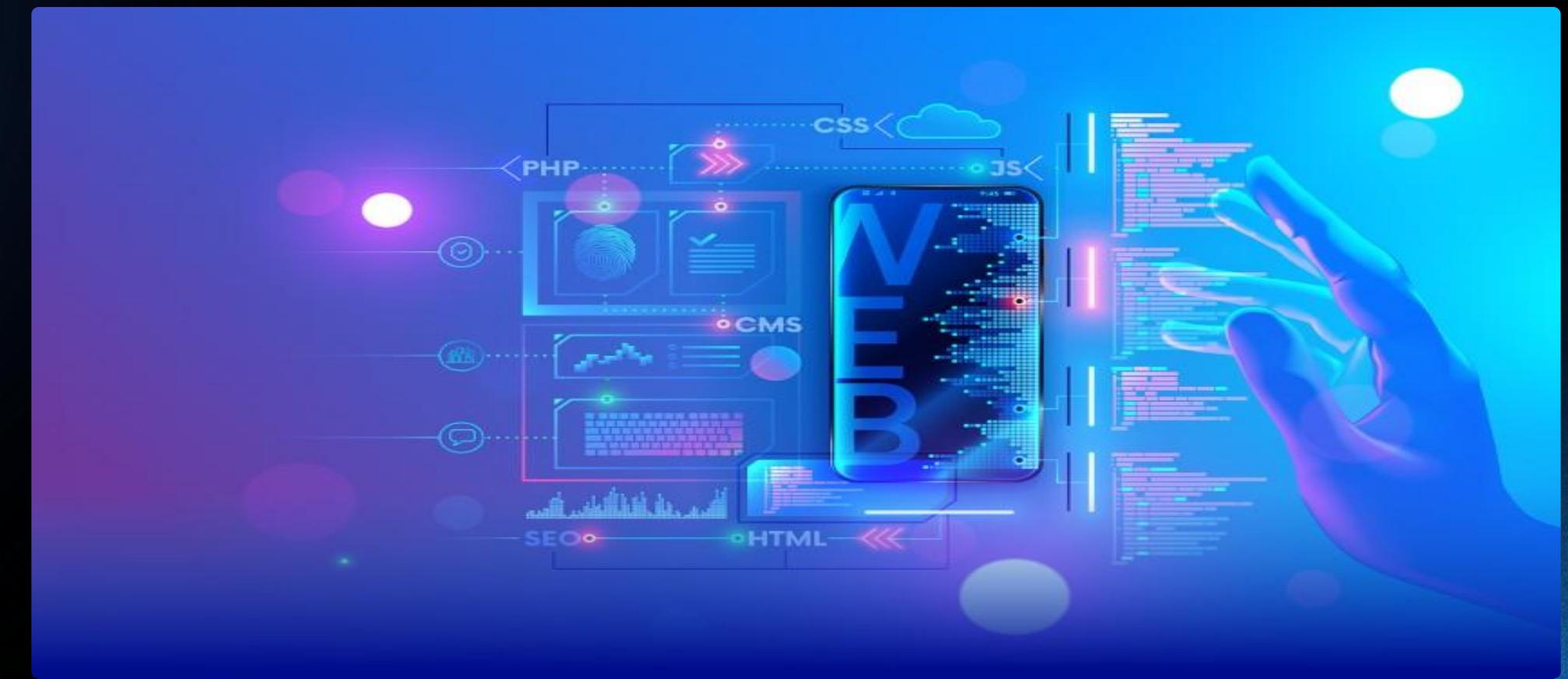
Comparación con otras estructuras de datos

Estructura	Claves Únicas	Acceso Rápido	Orden Garantizado
Listas	✗	✗	✗
Objetos	✓	✗	✓
Map	✓	✓	Depende del tipo



CONCLUSIÓN

- El uso de un Map en programación orientada a objetos es esencial para organizar y acceder a los datos de manera eficiente. 
- Ya sea que necesites acceso rápido a valores, ordenar tus datos o mantener el orden de inserción, hay una implementación de Map adecuada para cada caso. 





Referencias

1. Oracle. (2018). Map (Java SE 8 & JDK 8). Recuperado de:
<https://docs.oracle.com/javase/8/docs/api/java/util/Map.html>
2. GeeksforGeeks. (2021). Map interface in Java with examples. Recuperado de:
<https://www.geeksforgeeks.org/map-interface-java-examples/>
3. W3Schools. (s/f). Java HashMap. Recuperado de:
https://www.w3schools.com/java/java_hashmap.asp

THANK YOU