Ballerina
Swan Lake

Mastering Microservices and APIs with Ballerina: A Practical Guide

Hello!

Thisaru Guruge

thisaru@wso2.com | Associate Technical Lead **|** **@ballerinalang** **|** **WSO2**

Dimuthu Madushan

dimuthum@wso2.com | Software Engineer **|** **@ballerinalang** **|** **WSO2**

About this Session

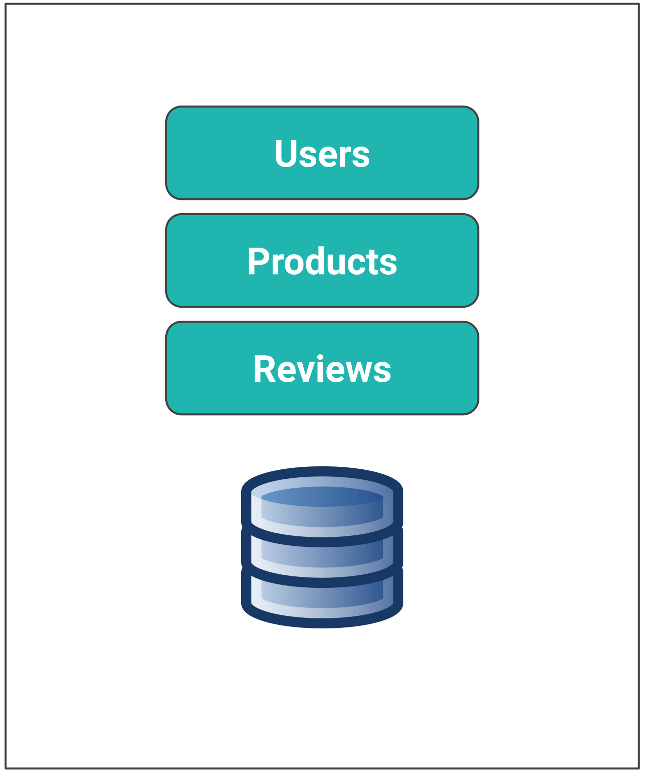
Ballerina
Swan Lake

Coming Up

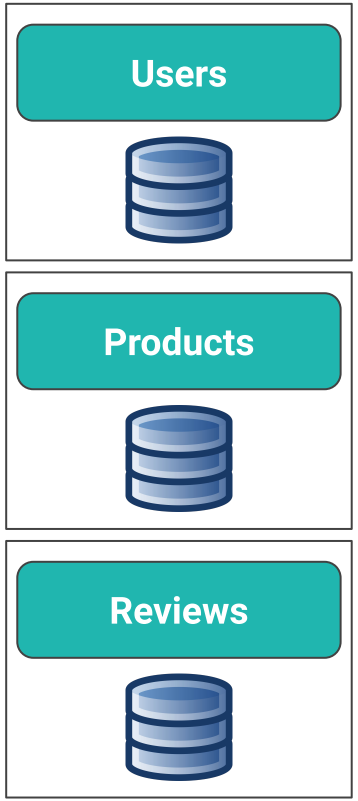
Introduction to Microservice Architecture

Ballerina
Swan Lake

Monolithic vs. Microservice Architecture



Monolithic Architecture



Microservice Architecture

Ballerina Swan Lake

Introduction to Microservices Architecture

- Characteristics of Microservices
    - Autonomous
    - Specialized
- Benefits of Microservices
    - Agility
    - Flexibility of scaling
    - Easy deployment
    - Technological freedom
    - Resilience

What is an API?

Introduction to Microservices Architecture

REST (**RE**presentational **S**tate **T**ransfer)

- Most widely used architectural style
- Uses the concept of resources
- Resources can be accessed via verbs and resource paths
- Each resource has a standard format to represent data; server sends - client understands

Introduction to Microservices Architecture

GraphQL

- Relatively new protocol developed by Facebook
- Fast adaptation from the major companies
- Query language for APIs
- Data is structured as a hierarchical structure
- Has a single endpoint
- Clients can request exactly what they want, server responds with exactly what was requested

Introduction to Microservices Architecture

Real-World Examples

- Amazon
- Netflix
- Uber
- Shopify
- Facebook

Understanding Ballerina Basics

Ballerina
Swan Lake

Understanding Ballerina Basics

- Language made specifically for integration and microservices
- Ready for cloud
- Built-in support for network endpoints
- Rich library - A set of packages to help writing and connecting to various endpoints
- Data types suitable for network communication

Understanding Ballerina Basics: Data Types in Ballerina

- **`int`**: Integer data type (32-bit signed integer)
- **`float`**: Floating-point data type (64-bit double-precision floating-point)
- **`decimal`**: Decimal data type for precise decimal arithmetic
- **`boolean`**: Boolean data type (true or false)
- **`string`**: String data type (a sequence of Unicode characters)
- **nil**: Ballerina's version of null is called nil and written as `()`
- **Union Types**: `T1|T2` is the union of the sets described by `T1` and `T2`

```
// Integer
int i = 10;


// Float
float f = 12.34;


// Decimal
decimal d = 12.34d;


// Boolean
boolean b = true;


// String
string s = "Hello World!";


// Nil
int? n = ();


// Union (either string or int)
string|int x = 10;
```

Ballerina
Swan Lake

Understanding Ballerina Basics: Data Types in Ballerina

- **Arrays**: An array can be used to hold a list of values of a given type
- **Maps**: The `map<T>` type is a data structure to store key-value pairs, with a `string` key and a value of a given type
- **anydata**: The umbrella type representing any data type
- **Other Types** : `table`, `stream`, `byte`, `error`, `enum`, etc..,

```
// Int array
int[] numbers = [73, 42, 6174];

// String array
string[] names = ["John", "Doe", "Jane", "Doe"];

// Int map
map<int> ages = {
    "John": 30,
    "Jane": 20,
    "Karen": 40
};

// anydata array
anydata[] data = [1, "hello", 3.4, true];
```

Ballerina
Swan Lake

Understanding Ballerina Basics: Data Types in Ballerina

- **Record**: A collection of specific named fields where each field has a type for its value

```
type Address record {
    int number;
    string street;
    string city;
};

type Profile record {
    string name;
    int age;
    Address address;
};
```

Ballerina
Swan Lake

Understanding Ballerina Basics: Data Types in Ballerina

- **JSON**: Used to send data over the network. Union of simple basic types `()|boolean|int|float|decimal|string|json[]|map<json>`
- **XML**: A markup language and file format for storing, transmitting, and reconstructing arbitrary data

```
json profile = {
    name: "John Doe",
    age: 30,
    address: {
        city: "Colombo",
        country: "Sri Lanka"
    },
    contacts: [
        {
            kind: "email",
            value: "john@example.com"
        },
        {
            kind: "phone",
            value: "+1-202-555-0105"
        }
    ]
};
```

Ballerina
Swan Lake

Understanding Ballerina Basics: Functions

- Functions are building blocks of an application
- The `function` keyword is used to define functions in Ballerina
- A function can have zero or more input arguments and can return a value (Not returning anything means returning nil)

```
function add(int a, int b) returns int {
    return a + b;
}
```

Ballerina
Swan Lake

Understanding Ballerina Basics: Hello World!

- Execute the `bal new hello_world` to create a new Ballerina project
- Code:

```
import ballerina/io;


public function main() {
    io:println("Hello, World!");
}
```

- The `main` function is the entry point of a Ballerina program
- Execute `bal run` to run the program

Understanding Ballerina Basics: Services

- The `service` and `listener` are built-in constructs in Ballerina
- They provide an easy way to write network endpoints that serves client requests
- Execute the `bal new hello_world_service` command to create a new Ballerina project

```
import ballerina/http;


service on new http:Listener(9090) {
    resource function get greeting() returns string {
        return "Hello, World!";
    }
}
```

Ballerina Swan Lake

Understanding Ballerina Basics: Clients

- The `client` is also a built-in construct in Ballerina
- Clients provide an easy way to consume services

```
import ballerina/http;
import ballerina/io;

public function main() returns error? {
    http:Client greetingClient = check new("http://localhost:9090");
    string greeting = check greetingClient->/greeting;
    io:println(greeting);
}
```

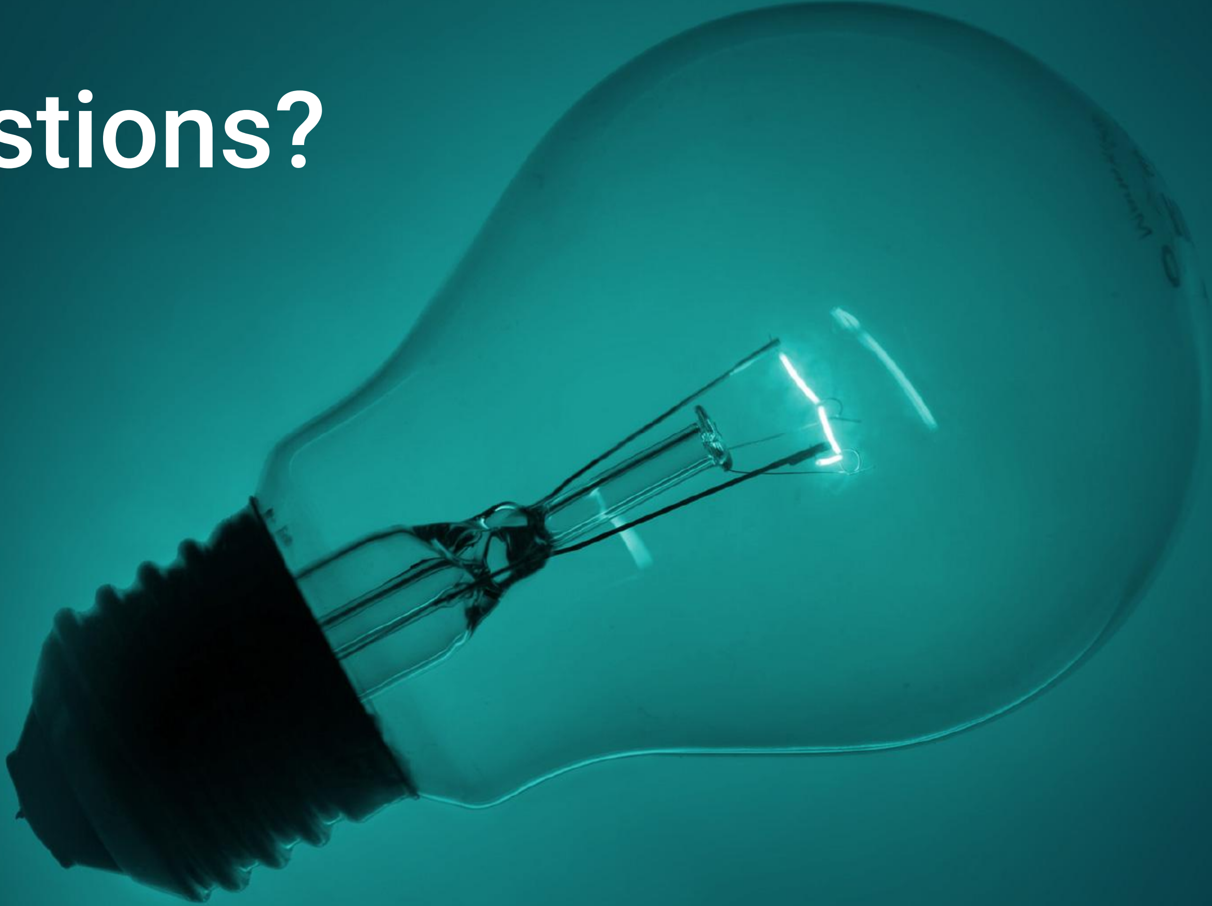
Ballerina Swan Lake

Activity

Write a Ballerina program to get all the repositories in ballerina-platform GitHub organization and star all the repos.

- Need a GitHub token with “repository” scope
- API: `https://api.github.com`
- Paths:
  - To retrieve repos: `GET` `/orgs/[org]/repos`
    - Repos per page: 100
    - Sort by “updated”
  - To star a repo: `PUT` `/user/starred/[org]/[repoName]`

Questions?



Ballerina
Swan Lake

Mini Project

- Do something cool with/about Ballerina
  - A new Ballerina package, published to Ballerina central
  - An article/video about Ballerina
  - Contribute to Ballerina project (Find "Good First Issues")

- Successful submissions will receive free vouchers for WSO2 practitioner and developer certifications.
  - Make sure your source code/article/video is public

- There's no limit, submit as many entries as you want

Ballerina Swan Lake

Fun Activity

- Connect to my service at `http://10.30.10.22:9090`
- Get the profiles using `GET /profiles`
- Sort the retrieved profiles by their age
- Send the sorted names array using `POST /submit`
- The request body should be:

```
{
    “name”: “Your Name”,
    “answer”: [“array”, “of”, “names”, “sorted”, “by”, “age”]
}
```

Ballerina
Swan Lake

Find out more...

- Learn Ballerina:
  - Ballerina By Example
    - https://ballerina.io/learn/by-example/
  - API Documentation
    - https://lib.ballerina.io/
  - Submit your mini projects here:
    - https://forms.gle/5K3VGr1irY44aLoU7

- Join the Ballerina community

Discord — ballerinalang

COLLECTIVES on stackoverflow — WSO2 Collective

X — @ballerinalang

GitHub — ballerina-lang

Thank you!

If you have any further questions, please raise them in the **Ballerina Discord server.**

https://discord.gg/ballerinalang

Ballerina
Swan Lake