# Application Server

Для того, чтобы написанные разработчиком эндпоинты работали в сети - принимали запросы, парсили их и возвращали ответы - требуется **сервер**, который бы постоянно прослушивал порты, запускал наши эндпоинты и составлял + отправлял http ответы обратно.

Такой сервер называется Application Server.

Этот сервер это такой же набор функций.

Поэтому его подключают как библиотеку и при запуске кода он начинает работать.

В пайтоне эту роль играет библиотка uvicorn или gunicorn. Они выполняют перечисленный выше функционал:

from fastapi import FastAPI

import uvicorn

app = FastAPI()

@app.get(‘/end\_one/’)

def end():

return ‘hello\_world’

if \_\_name\_\_ = “\_\_main\_\_”:

uvicorn.run(port = 8080:8080, …)

В Java это Tomcat, GlassFish, WebLogic

## Servlet Container / Full Java EE Server

Servlet Container - сервер который поддерживает только минимальный набор форматов, которые оперируются в эндпоинтах (Tomcat)

Full Java EE Server - сервер который поддерживаем много различных форматов (GlassFish, WebLogic)

Форматы:

| **Компонент** | **Назначение** |
| --- | --- |
| **Servlet API** | Работа с HTTP |
| **JSP** | Шаблоны для HTML |
| **JPA** | Работа с БД (ORM) |
| **JAX-RS** | REST API |
| **EJB** | Бизнес-логика |
| **JMS** | Сообщения (message queue) |
| **CDI** | Инъекция зависимостей |
| **JTA** | Транзакции |

# Фреймворк для написания эндпоинтов

Весь рутинный функционал реализации эндпоинтов обёрнут в интерфейс, предоставляемый фреймворками (библиотеками)

В пайтон это например flask, FastAPI

from fastapi import FastAPI

app = FastAPI()

@app.get(‘/end\_one/’)

def end():

return ‘hello\_world’

В Java это библиотека Jakarta EE. И это даже больше чем библиотека - это стандарт, которые другие разработчики должны учитывать, чтобы их библиотеки и др считались jakarta совместимыми. Так сложилось из за крайней популярности jakarta/

package com.example;

import jakarta.ws.rs.GET;

import jakarta.ws.rs.Path;

import jakarta.ws.rs.Produces;

import jakarta.ws.rs.core.MediaType;

@Path("/hello")

public class HelloResource {

@GET

@Produces(MediaType.APPLICATION\_JSON)

public String hello() {

return "{\"message\":\"Hello, world!\"}";

}

}