Making CoAP.NET great again!

Is it worth to await?

Philip Wille

Introduction

- Synchronous and asynchronous execution

Introduction

- Synchronous and asynchronous execution
- **T**ask-based **A**synchronous **P**attern (TAP)

Introduction

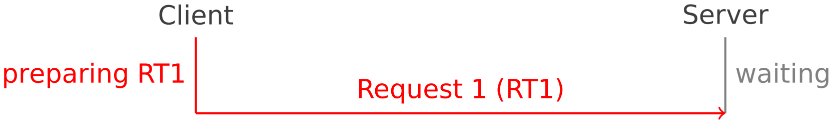
- Synchronous and asynchronous execution
- **T**ask-based **A**synchronous **P**attern (TAP)
- **Co**nstrained **A**pplication **P**rotocol (CoAP)

Write synchronous code in C#

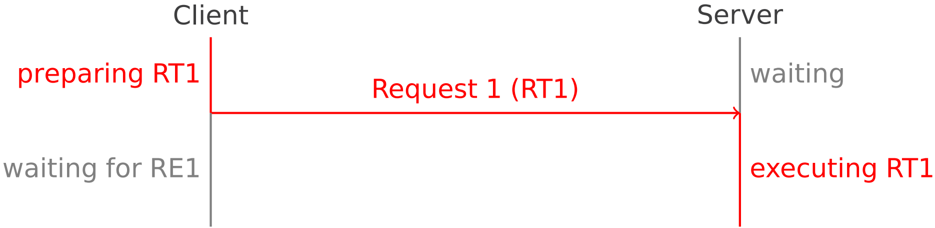
```
public List<Person> GetAll()
{
    var persons = this.context.Persons.ToList();
    return persons;
}

public void PrintPersons()
{
    foreach (var person in persons.GetAll())
    {
        Console.WriteLine(person);
    }
}
```

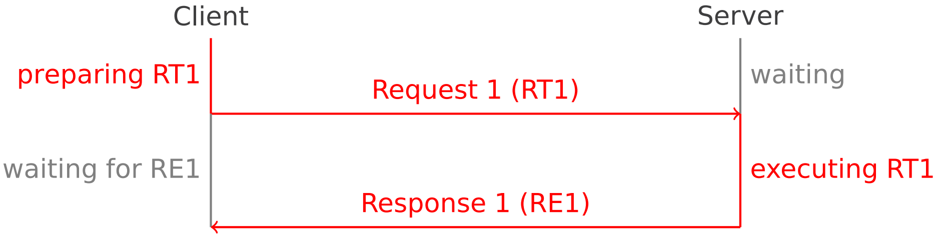
Synchronous execution



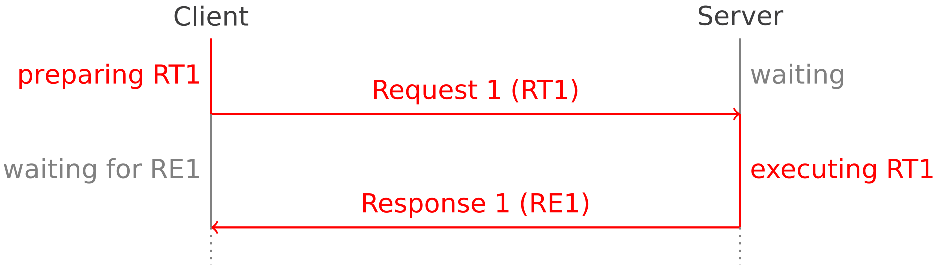
Synchronous execution



Synchronous execution



Synchronous execution



Client

Server

preparing RT1

waiting

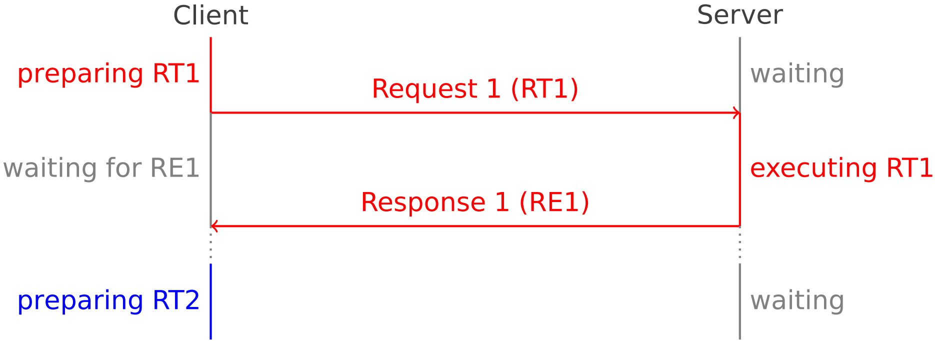
Request 1 (RT1)

waiting for RE1

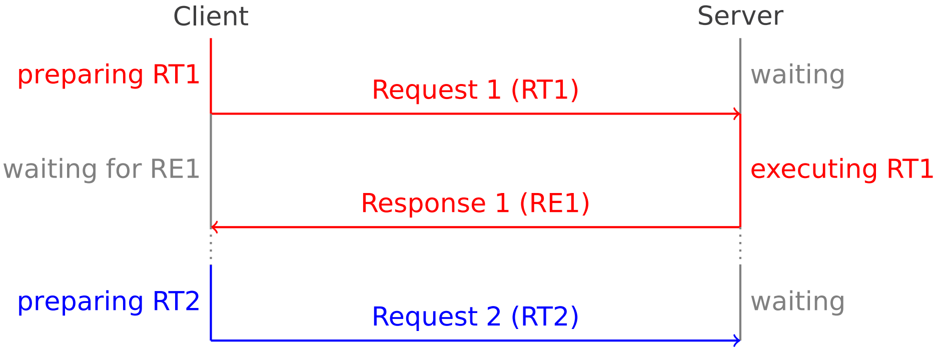
executing RT1

Response 1 (RE1)

Synchronous execution



Synchronous execution



Client

Server

preparing RT1

waiting

Request 1 (RT1)

waiting for RE1

executing RT1

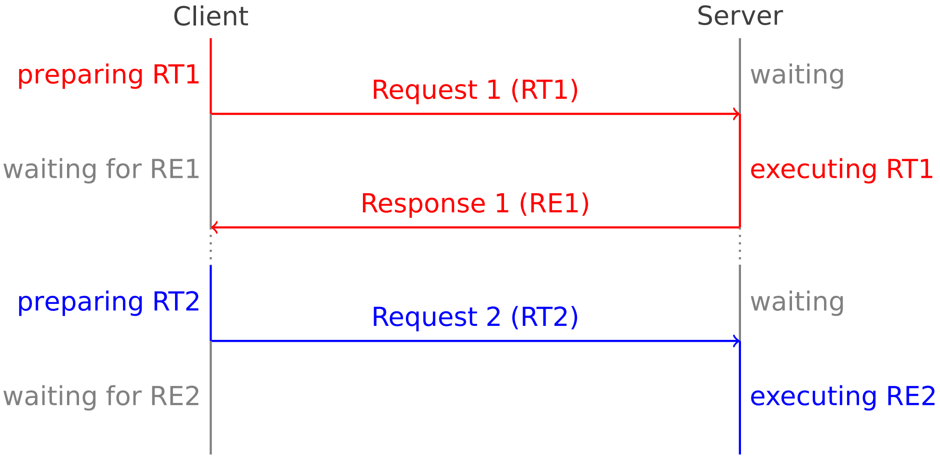
Response 1 (RE1)

preparing RT2

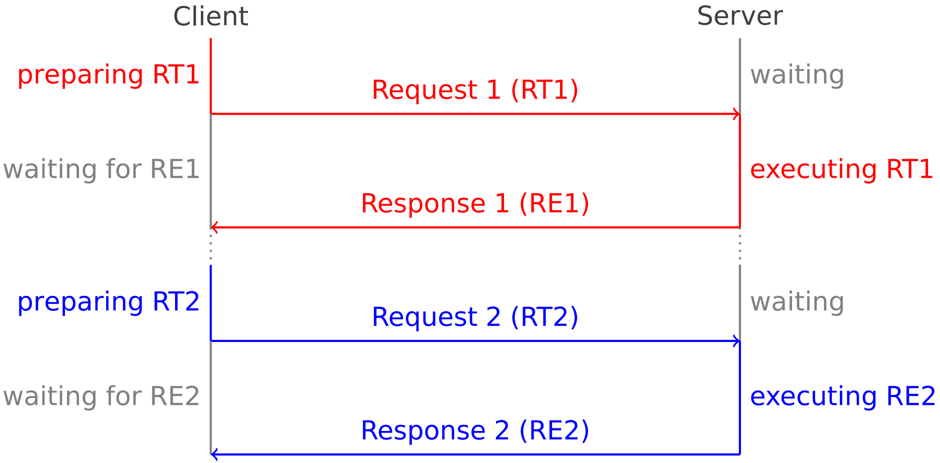
waiting

Request 2 (RT2)

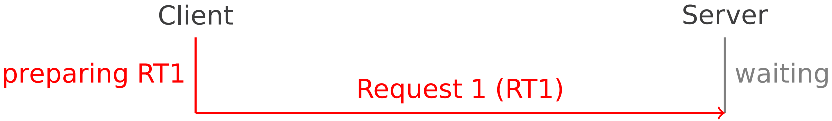
Synchronous execution



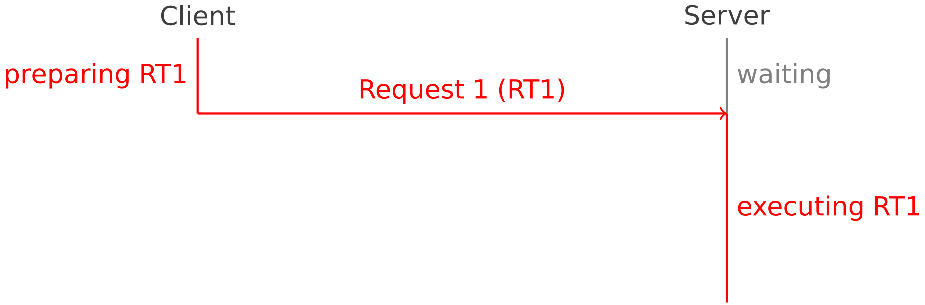
Synchronous execution



Asynchronous execution



Asynchronous execution



Client

Server

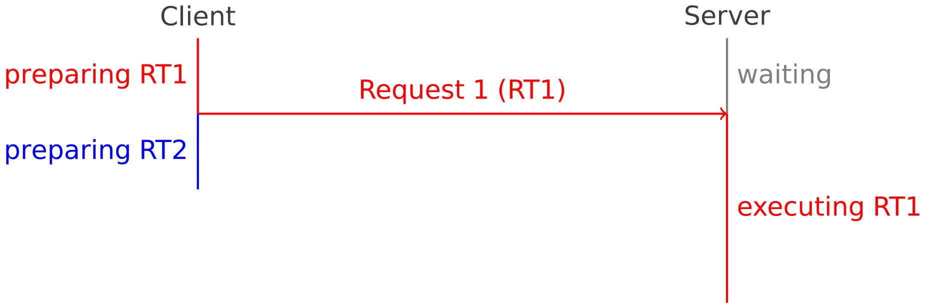
preparing RT1

waiting

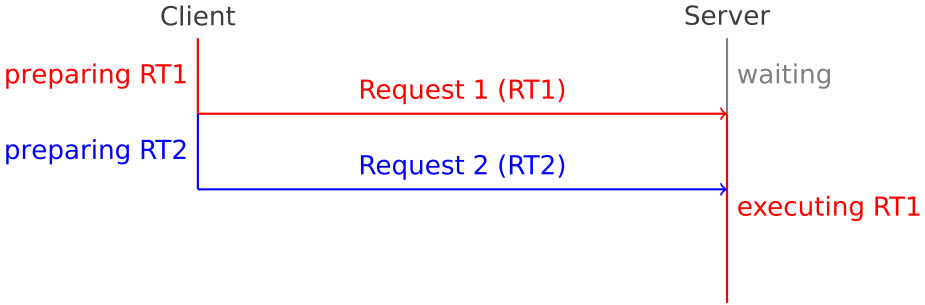
Request 1 (RT1)

executing RT1

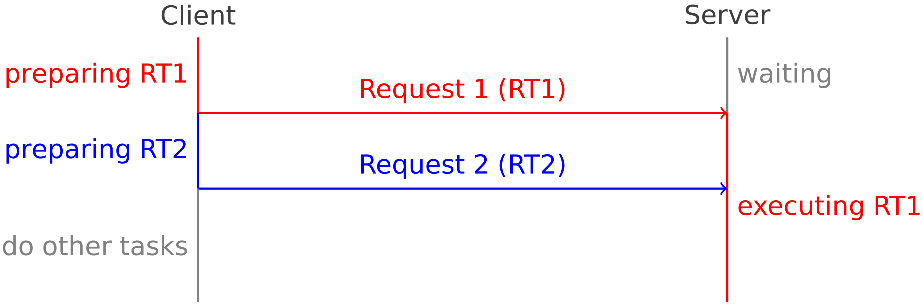
Asynchronous execution



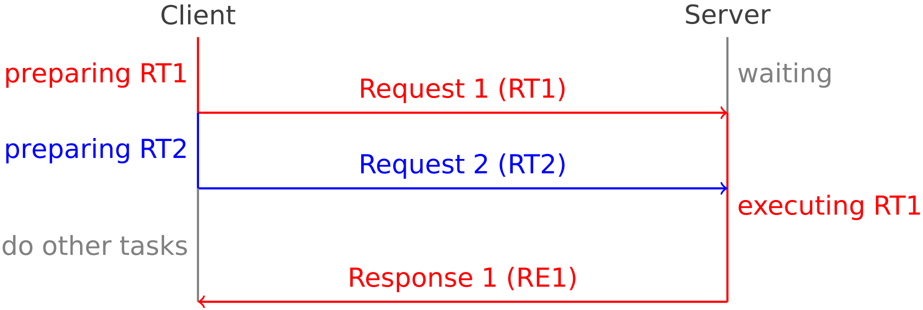
Asynchronous execution



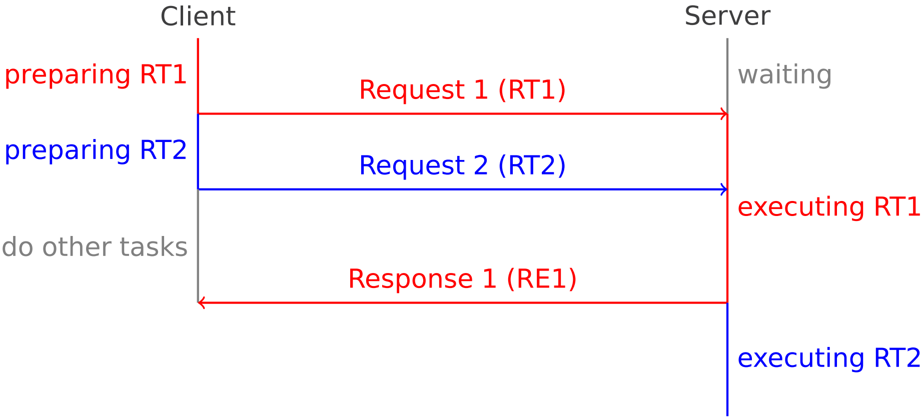
Asynchronous execution



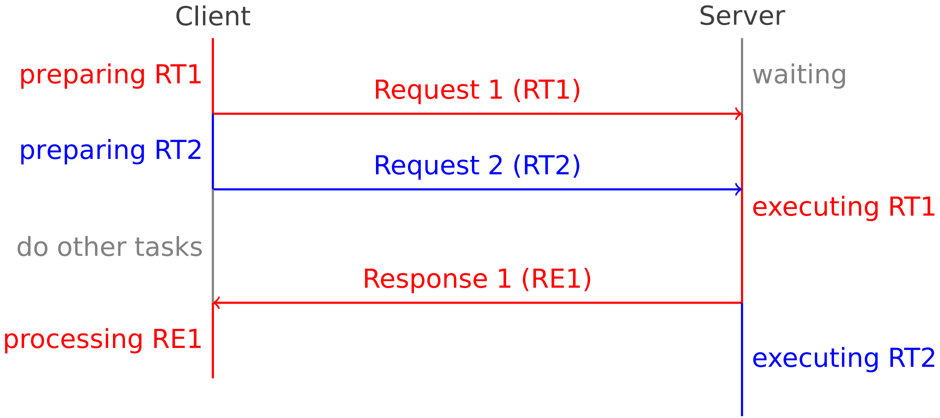
Asynchronous execution



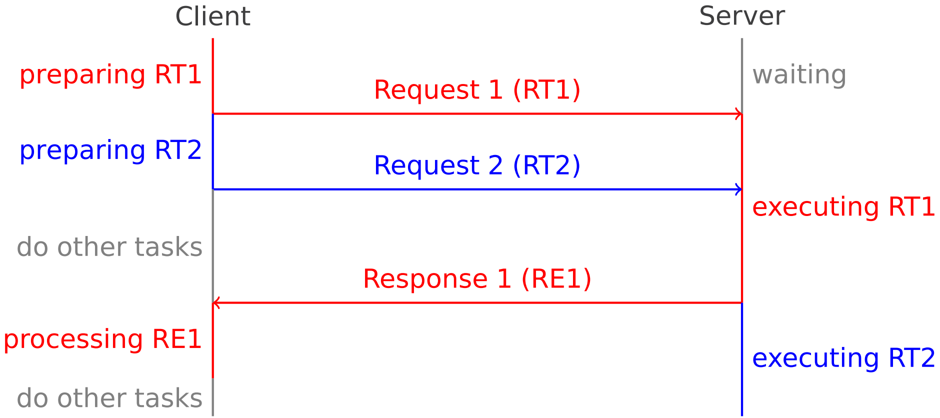
Asynchronous execution



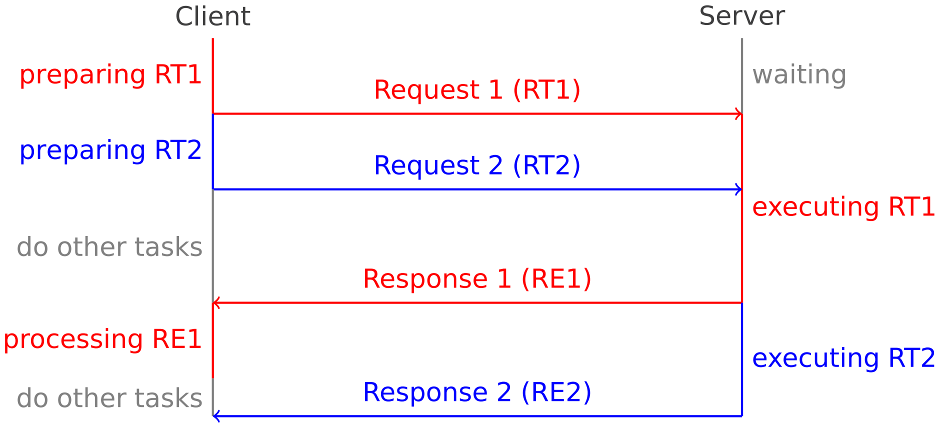
Asynchronous execution



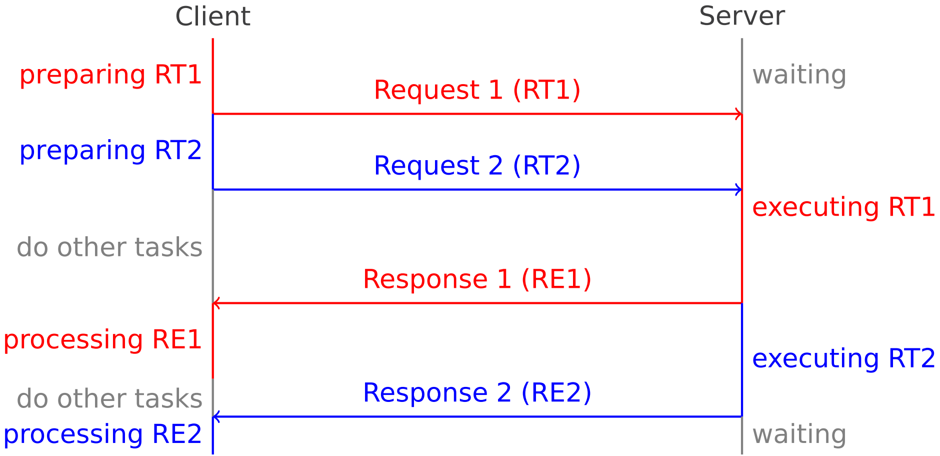
Asynchronous execution



Asynchronous execution



Asynchronous execution



Write asynchronous code in C#

```
public Task<List<Person■ GetAllAsync(CancellationToken cancellationToken)
{
    var persons = this.context.Persons.ToListAsync(cancellationToken);
    return persons;
}

public async Task PrintPersonsAsync(CancellationToken cancellationToken)
{
    var persons = await persons.GetAllAsync(cancellationToken).ConfigureAwait(false);
    foreach (var person in persons)
    {
        Console.WriteLine(person);
    }
}
```