# Playframework, Realtime Web

Sadek Drobi
Earle Casteldine

ZENEXITY

# Web Oriented Architectures

# Where are we now?

Exciting time to be in software development

Cloud, Distributed Computing, Data, Bigger Data, Big Data, Infinite Data, Functional Programming

...

# Object/Entity Oriented Programming

An object represents a business entity

Using getters and setter on that object manipulates database rows automatically

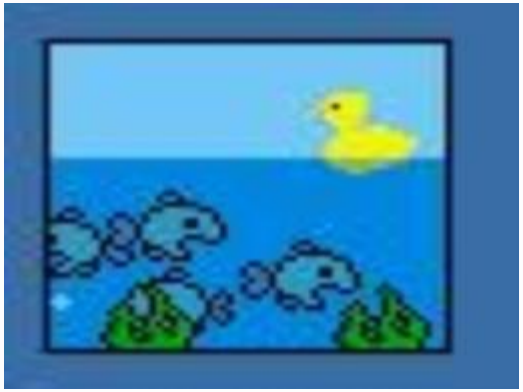
Enough to learn an ORM/Active Record to make a living

Very little problem solving

# Object/Entity Oriented Programming

# Then we rediscovered the importance of Data!

# Object/Entity Oriented Programming



# Object/Entity Oriented Programming

# Object/Entity Oriented Programming

# Object/Entity Oriented Programming

# New Era with new challenges

Analyzing big sets of data

Using several machines for computing and analyzing data

Integrating different sets of data in different formats

Handling lots of users requests

# Playframework

The Play framework makes it easier to build web applications with Java & Scala.

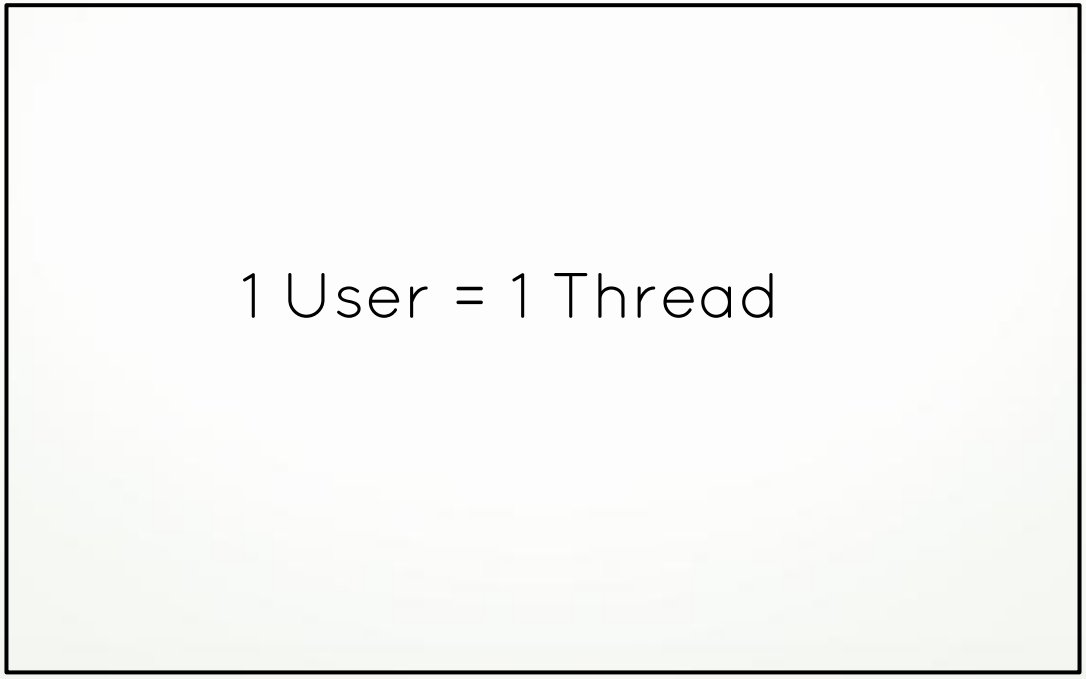
Play is based on a lightweight, stateless, web-friendly architecture for highly-scalable and realtime web applications

- thanks to its reactive model, based on Futures and Iteratee IO.

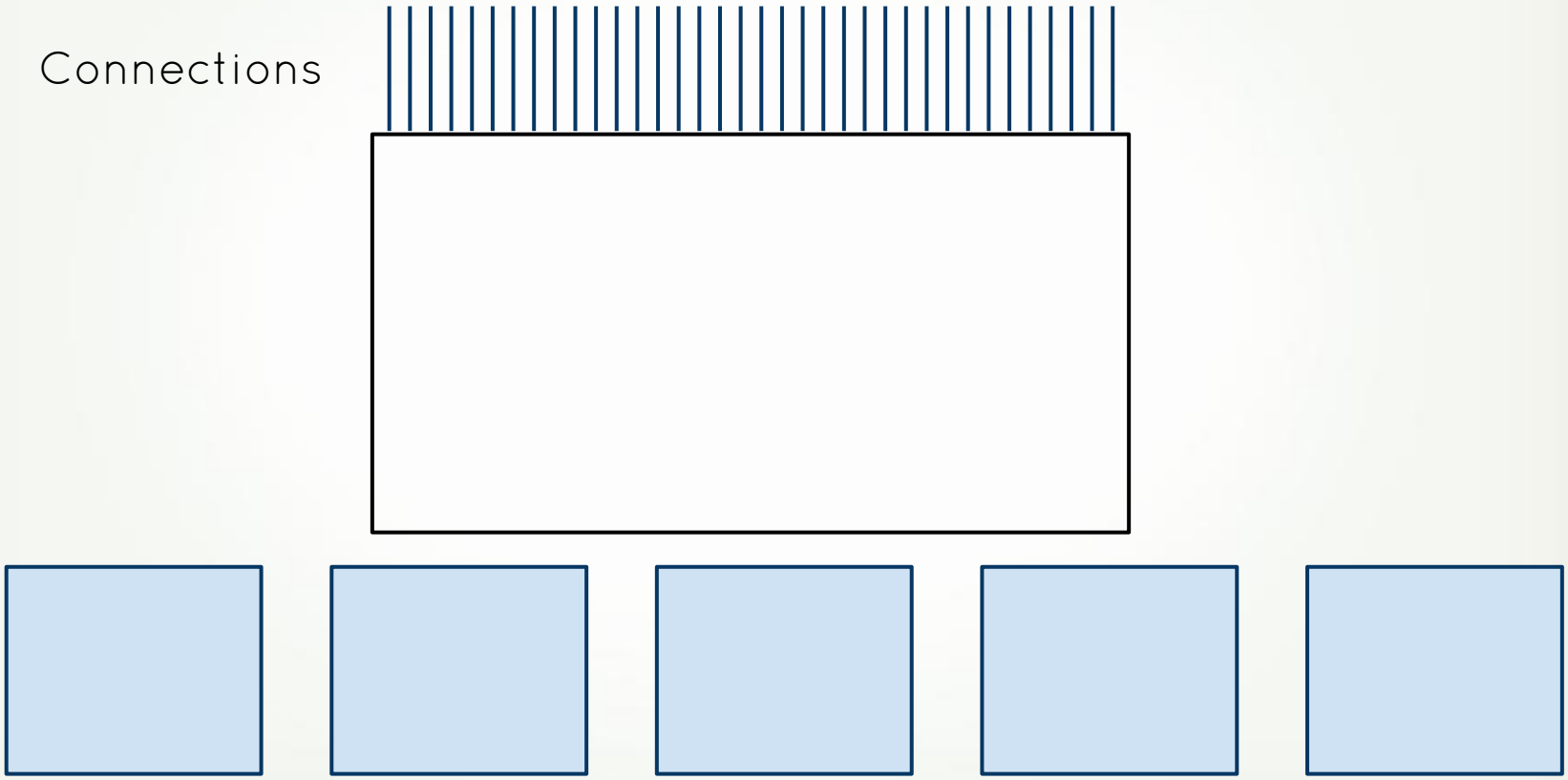
Play was born to live in the clouds.

# Demo

# Why another web framework?



# Why another web framework?



# New Era with new challenges

Analyzing big sets of data

Using several machines for computing and analyzing data

Integrating different sets of data in different formats

Handling lots of users requests

# Playframework, for a new era with new challenges

Analyze big sets of data: Functional Programming, Collections, Data Structures

Using several machines for computing and analyzing data

Integrating different sets of data in different formats: Parsers, APIs and FP

Handling lots of users requests: Stateless Reactive Architecture

# Web apps integrating Distributed Computing

Use distributed computing

Aggregate results back as a single response to the user

Programming Model

# Scheduled Computations, Computations in the Future

How do we talk about these computations?

How do we manipulate them?

# Scheduled Computations, Computations in the Future

Future[T]

# Scheduled Computations, Computations in the Future

# A Future

Represents a value that will be available in the Future (or a Failure)

Provides a way to be notified whenever the value is available (onComplete)

Has some nice composition properties

# A Future, nice properties

Future[Future[A]] => Future[A]

List[Future[A]] => Future[List[A]]

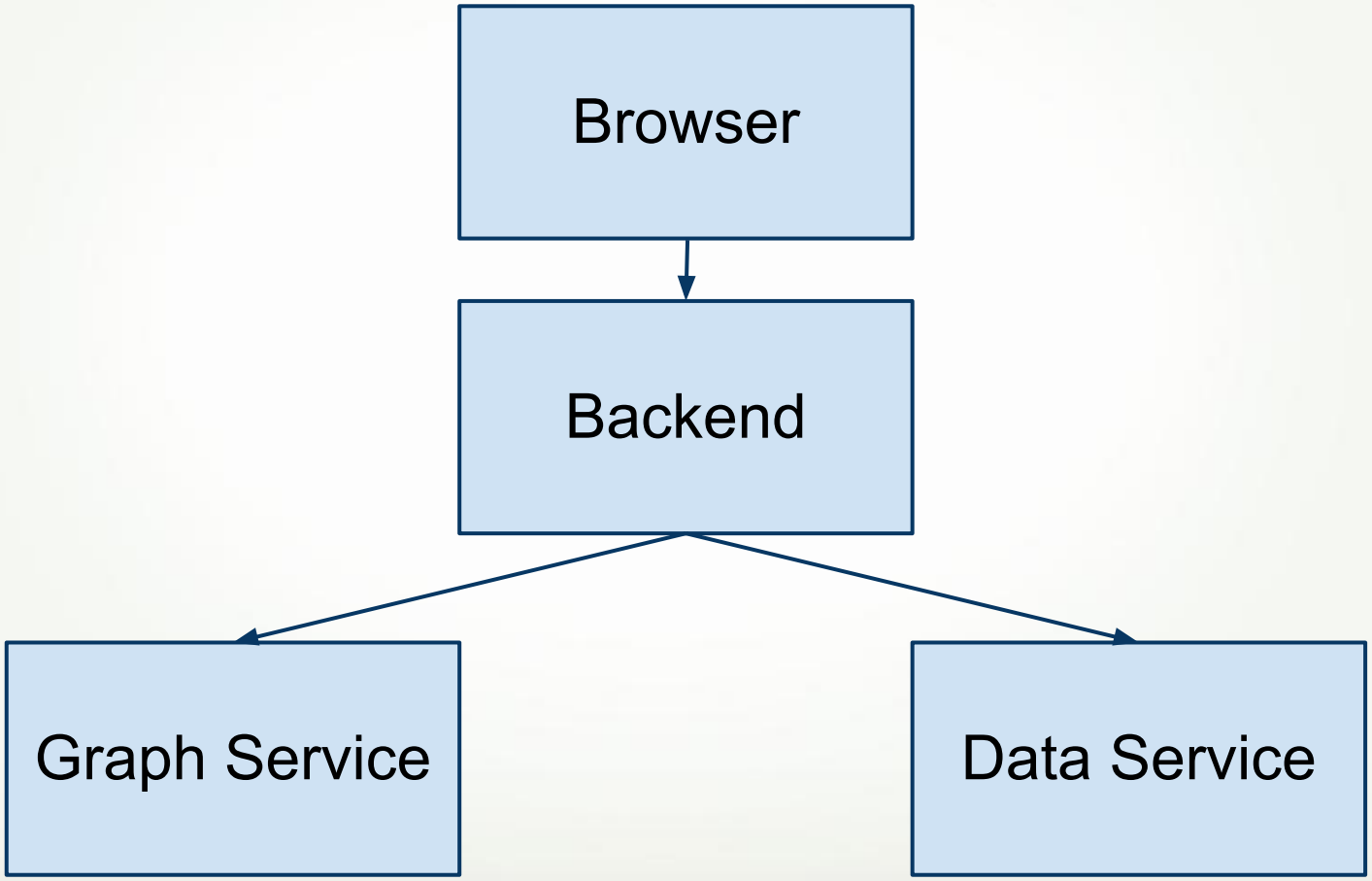
Future[A] => (A => B) => Future[B]

And a nice api with map, flatMap, filter, collect, recover, either and other higher order functions

# Using Futures for aggregating distributed computations

Context: Productivity tool used for discovery through data visualization

# Using Futures for aggregating distributed computations



# Demo

# Using Futures for aggregating distributed computations

waitAll kind of composition

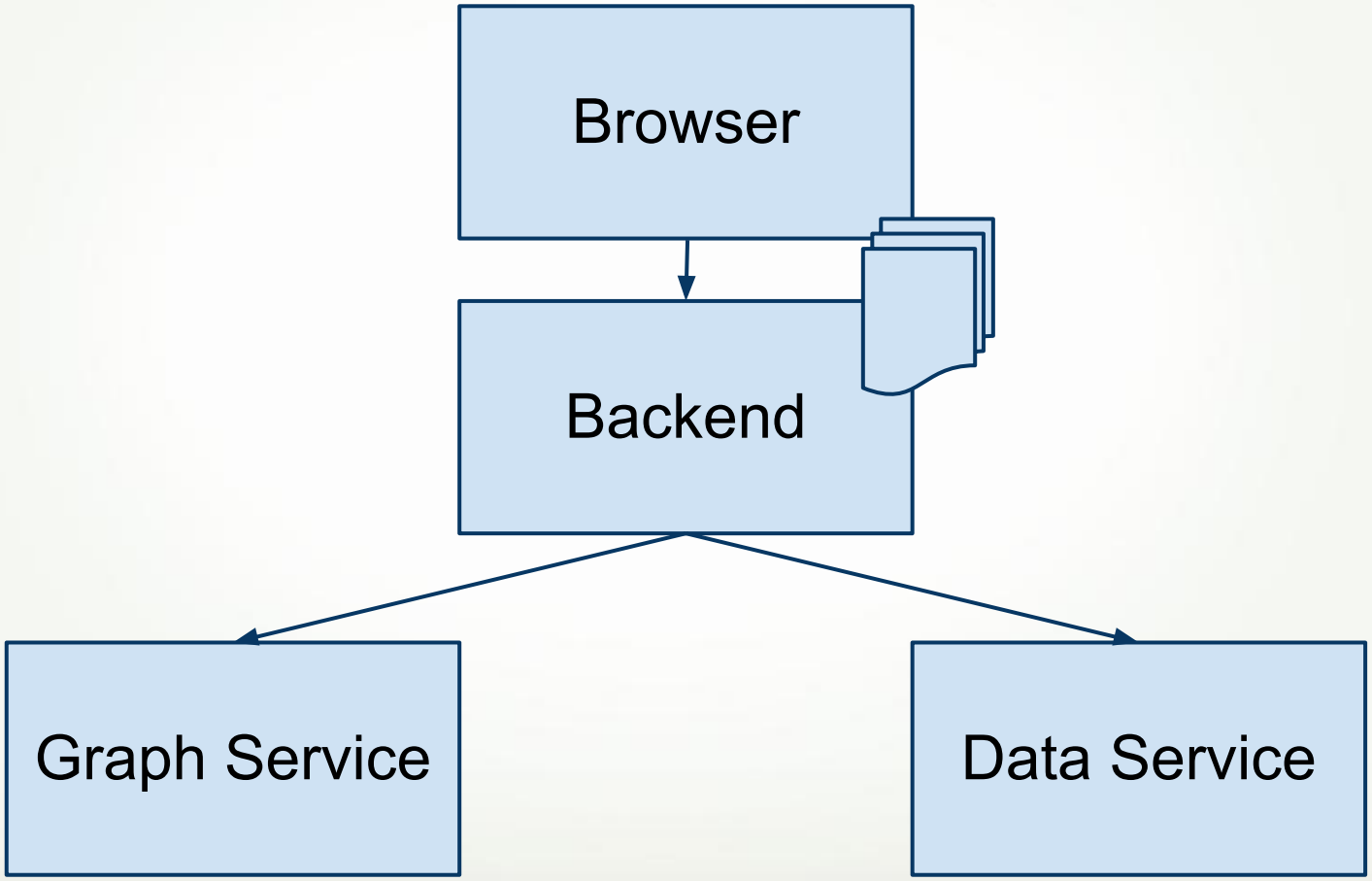
We can do waitAny, waitEither, timeout, ...etc

You can compose all of these

# Using Futures for aggregating distributed computations

Context: Productivity tool used for discovery through data visualization

# Aggregating and streaming distributed computations



# Aggregating and streaming distributed computations

We need a protocol to send messages to the browser

We need a way to talk about Streams

How can we transform our list of computations into a Stream?

# On the browser

Comet, Server Sent Events, Websockets.

# How can we talk about Streams at the server side?

# Is this what we really need to deal with Streams?

```
    });

    // Send a single 'Hello!' message
    out.write("Hello!");

}
```

# How can we talk about Streams on the server side?

Programming model

With nice properties

Compositional

# InputStream / OutputStream

But they lack composition properties, and they are blocking.

Did the industry move backwards?

# Iteratee[E,R]

Consumes chunks of type E

Eventually computes a value of type R

Fancy fold, you can pause!

# Iteratee[E,R]

Immutable state machine

Done(E,R)

Cont(E => Iteratee[E,R])

Error

# Iteratee[E,R], nice properties

Iteratee[E,Iteratee[E,R]] => Iteratee[E,R]

Future[Iteratee[E,R]] => Iteratee[E,R]

Iteratee[E,R] => (R => U) => Iteratee[E,U]

# Enumerator[E]

Iteratee[E,R] => Iteratee[E,R]

# Enumerator[E], nice properties

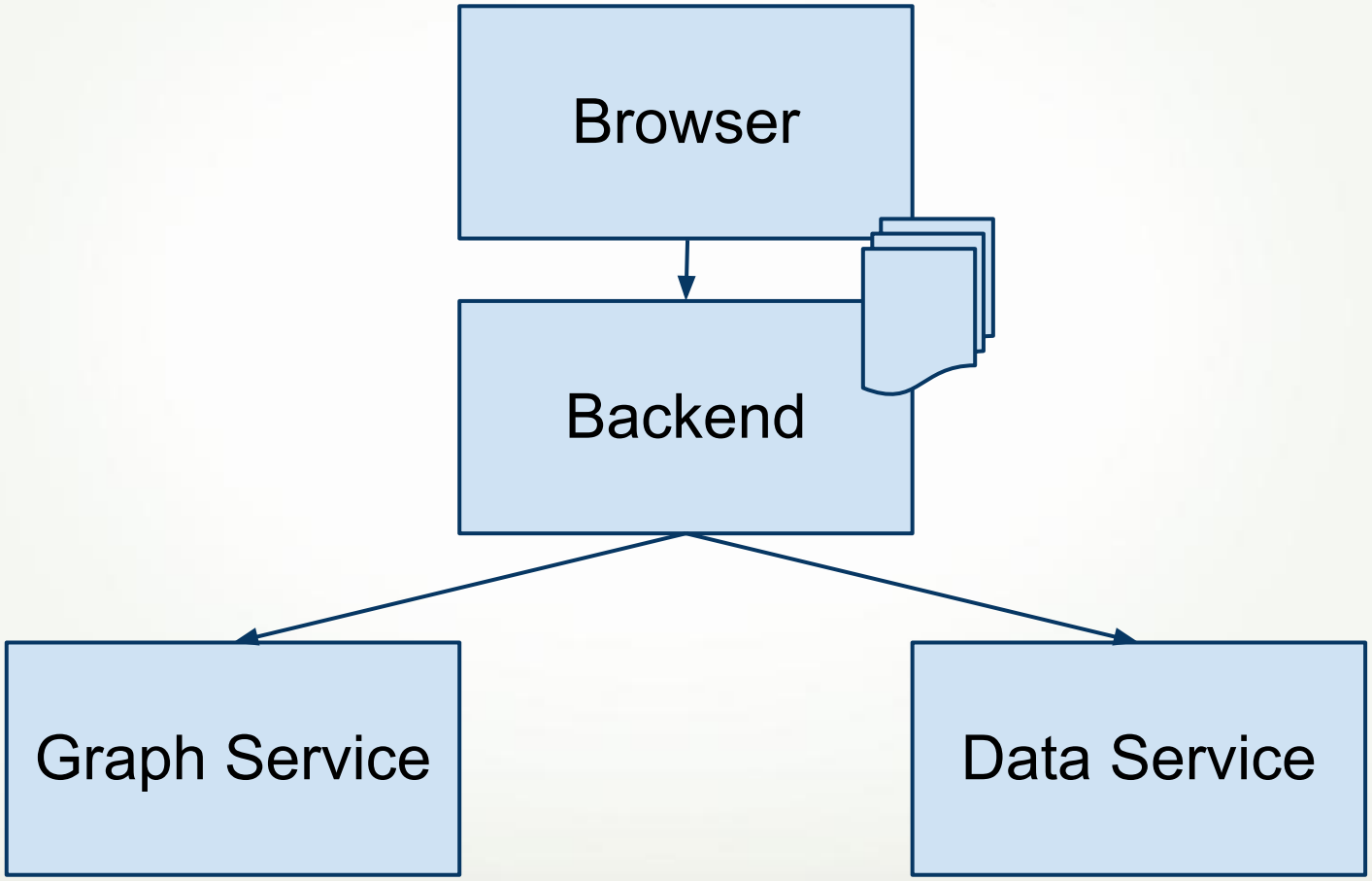
Enumerator[Enumerator[E]] => Enumerator[E]

Future[Enumerator[E]] => Enumerator[E]

and some nice functions

map, flatMap, interleave, compose, ...

# Aggregating and streaming distributed computations



# Demo

# Enumerator[E], Iteratee[E,R]

take, takeWhile, drop, dropWhile, buffer, collect, filter, map, scan, ...

# Infinite Data

Live streams of realtime Data

Queues, Storm, ...

With Iteratees, Enumerators, and Enumeratees

# Demo

# Questions?

- www.playframework.com
- Check and play with the provided samples
- Ask