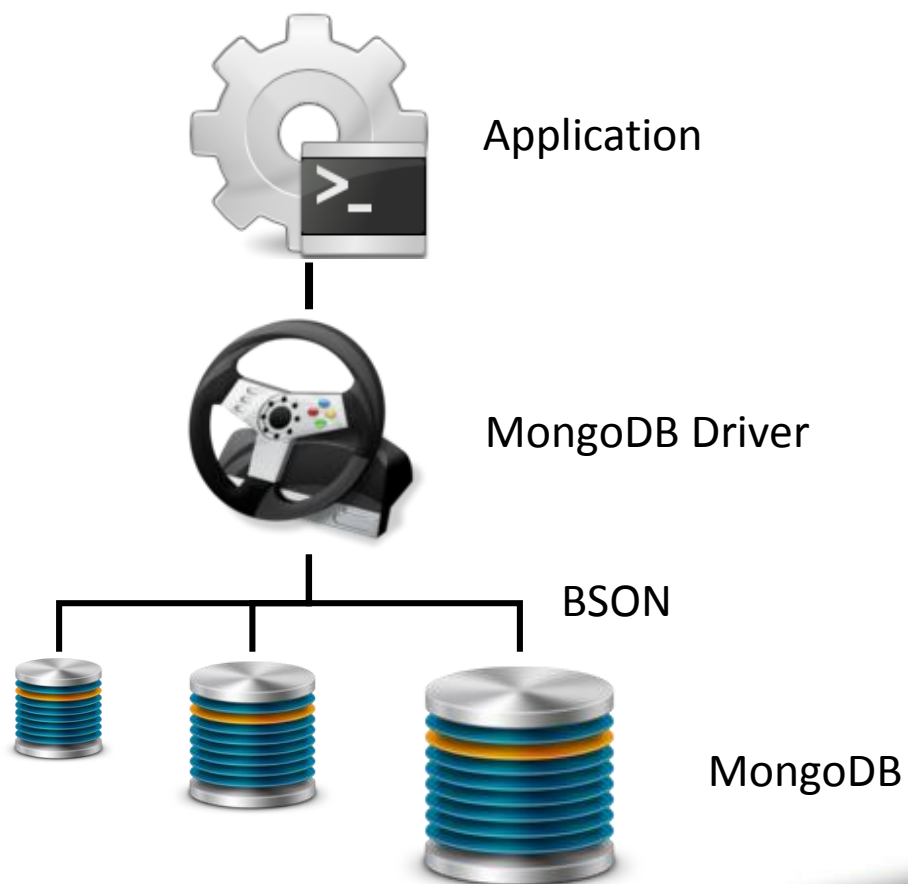


MongoDB C# драйвър

Да преговорим



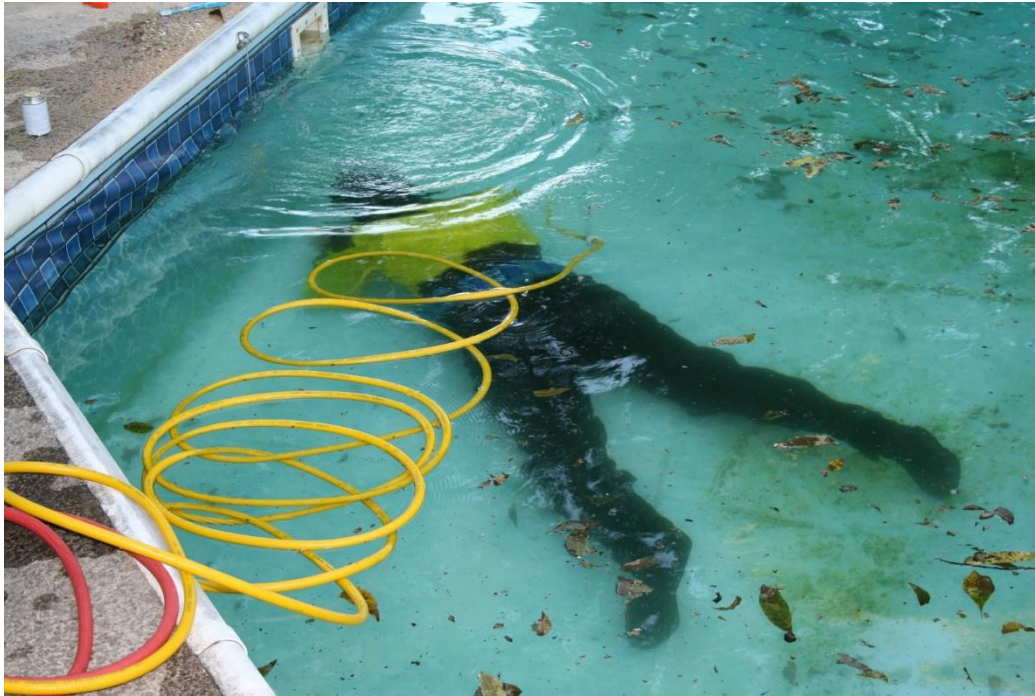
Проблеми

- Сериализация/Десериализация
- Съпоставяне на обекти и документи
- Типове
- Без схема?

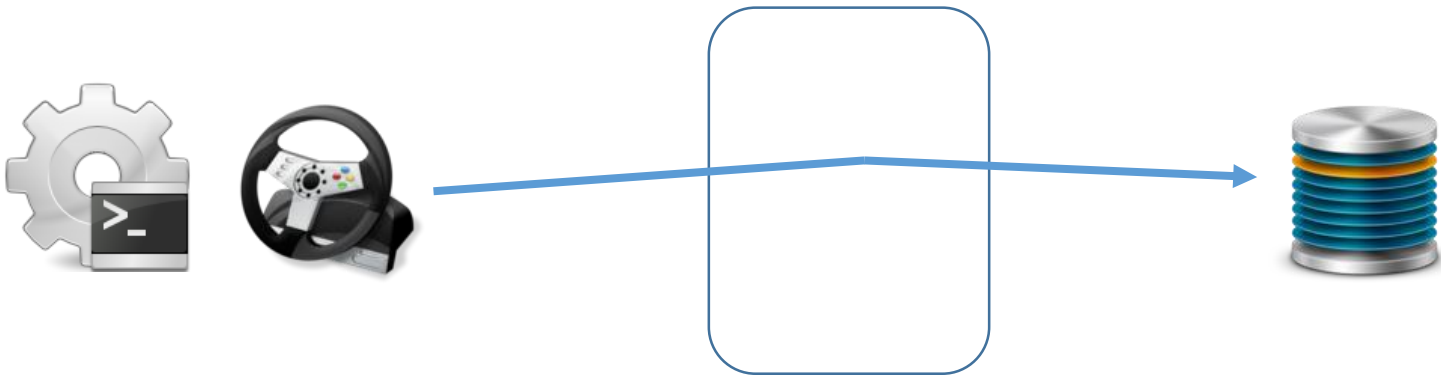
C# драйвър

- Лесен за използване
- Работи почти “магически”
- а.k.a. софтуер от щастливи програмисти
- Connection pool – не затваряйте ръчно връзки
- Прост **Object-Document Mapper (ODM)**
- **Language-Integrated Query (LINQ)**

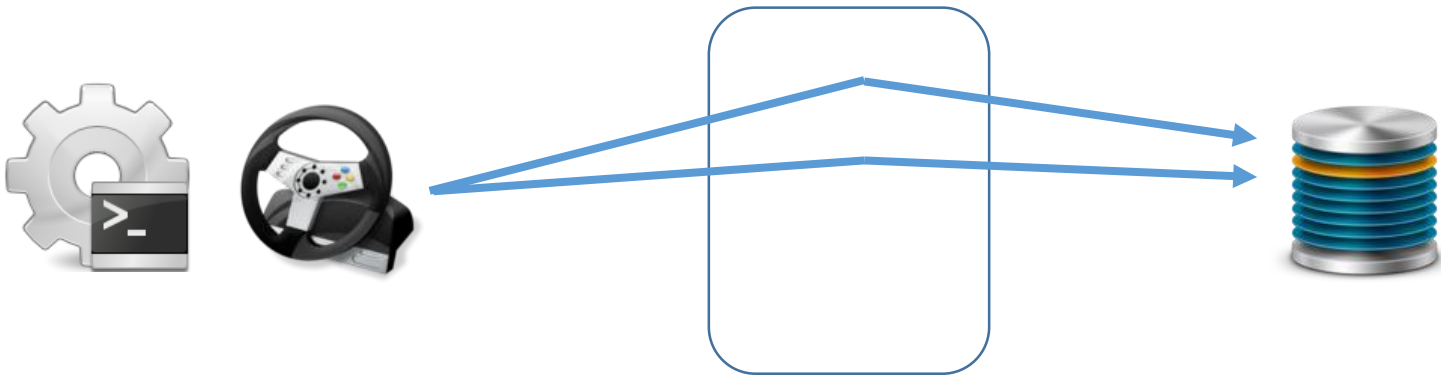
Connection pool ?!?



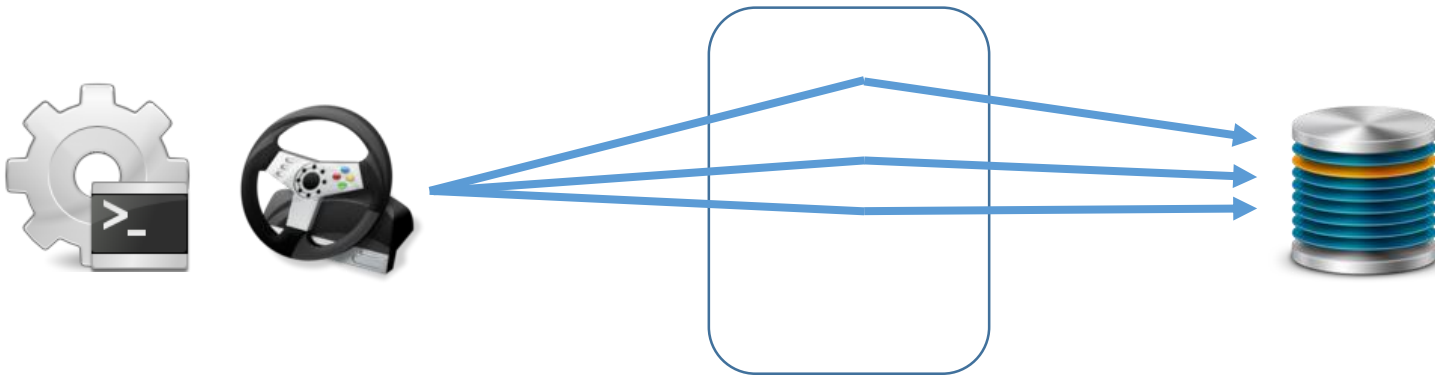
Connection pool



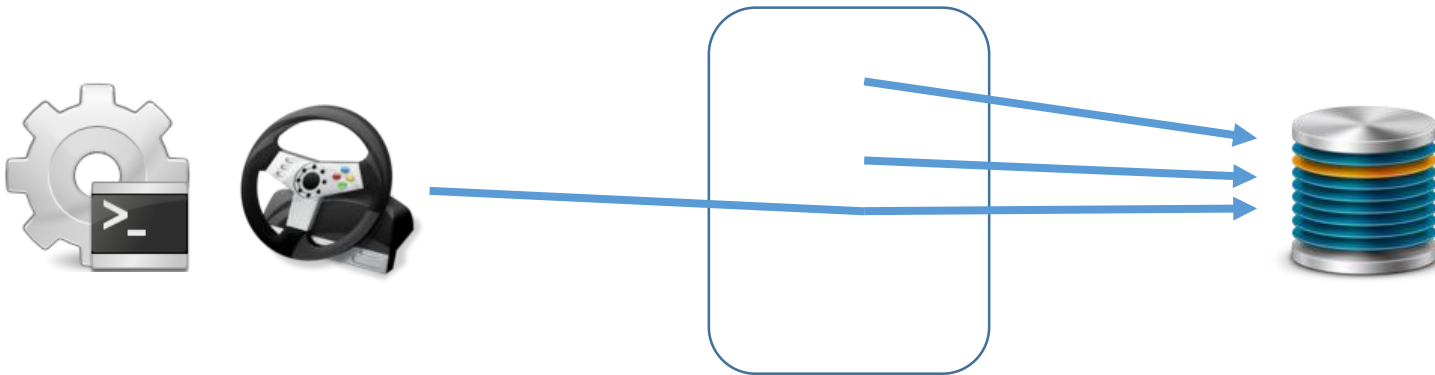
Connection pool



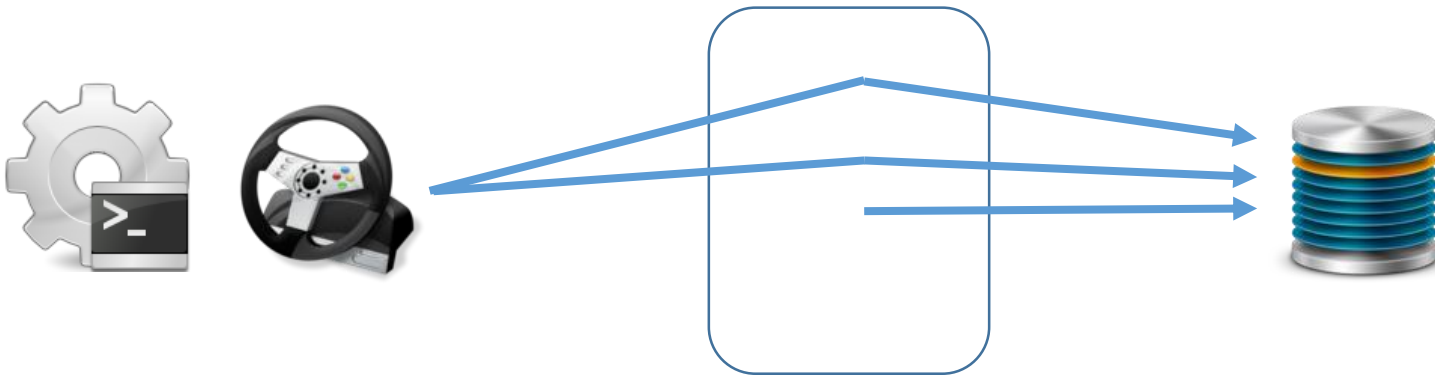
Connection pool



Connection pool



Connection pool



ODM


```
class Task
{
    string id;
    string title;
    DateTime date;

    int priority;

    IList<Comment> comments;
}

class Comment
{
    string author;
    string text;
}

{
    "_id": ObjectId("..."),
    "title": "...",
    "date": DateTime(...),
    "priority": 1,
    "comments": [
        {
            "author": "John",
            "text": "cool"
        },
        {
            "author": "Doe",
            "text": ":)"
        }
    ]
}
```



Инсталация

- Чрез NuGet – пакетен мениджър за .NET
 - Вграден във Visual Studio 2010+
 - "[mongocsharpdriver](#)"
 - Става с 3 клика
- Ръчно - като свалите .msi или .zip
 - Не си заслужава усилията



Клиент и Сървър

```
MongoClient client = new MongoClient("mongodb://localhost");  
MongoServer server = client.GetServer();
```

- Автоматично създава връзка
- Има и метод `server.Connect()`
 - няма нужда да го използвате

База от данни и колекция

```
MongoDatabase      database = server.GetDatabase("calendar");
```

```
MongoCollection      tasks = database.GetCollection("tasks");
```

```
MongoCollection<Task> tasks = database.GetCollection<Task>("tasks");
```

- Опционално задаване на клас за документите в колекция
- При разнотипни документи – може да се зададе при извличането им
- При миш-маш - може и анонимни класове

Добавяне

```
collection.Insert(new { title = "...", ... });
```

```
collection.Insert(new Task(...));
```

```
collection.Insert<OtherType>(new OtherType(...));
```

- Сериализира публичните полета
- Първото е обект от анонимен (динамичен) клас

Търсене

```
collection.FindAllAs<ExpandoObject>();
```

```
collection.FindAll();
```

```
collection.FindAllAs<OtherType>();
```

- Десериализира публичните полета
- ExpandoObject е клас с динамични полета
- Резултатът е курсор – имплементира IEnumerable
- Класът трябва да има конструктор без параметри

Търсене

```
var query = Query.Or(  
    Query.EQ("title", "Buy lemons"),  
    Query.EQ("title", "Make lemonade"),  
    Query.GT("priority", 2)  
);  
  
var cursor = collection.Find(query);  
  
foreach (var task in cursor) {  
    // ...  
}
```

Търсене

```
var cursor = collection.Find(query)
    .SetSortOrder("date", "priority")
    .SetSkip(1)
    .SetLimit(5);
```

- Внимание! **SetSortOrder** vs **OrderBy**
 - **SetSortOrder** сортира на сървъра
 - **OrderBy** сортира на клиента
 - Същото за **SetSkip** и **Skip**

Търсене

```
var document = collection.FindOne(query);
```

LINQ

```
var cursor =  
    from task in collection.AsQueryable<Task>()  
    where task.priority >= 1  
    orderby task.priority descending  
    select task;  
  
foreach (var task in cursor)  
{  
    // ...  
}
```

Добавяне или промяна

```
collection.Save(task);
```

- Еквивалента на upsert
 - Ако task съдържа id – ще го промени
 - Ако не – ще го добави

Промяна

```
collection.Update(  
    Query.EQ("_id", new ObjectId("...")),  
  
    Update.Combine(  
        Update.Push("tags", "new tag"),  
        Update.Set("date", DateTime.Now)  
    ),  
  
    UpdateFlags.Multi  
);
```

Изтриване

```
collection.Remove(  
    Query.EQ("_id", new ObjectId("...")),  
);  
  
collection.RemoveAll();
```

Демо тайм