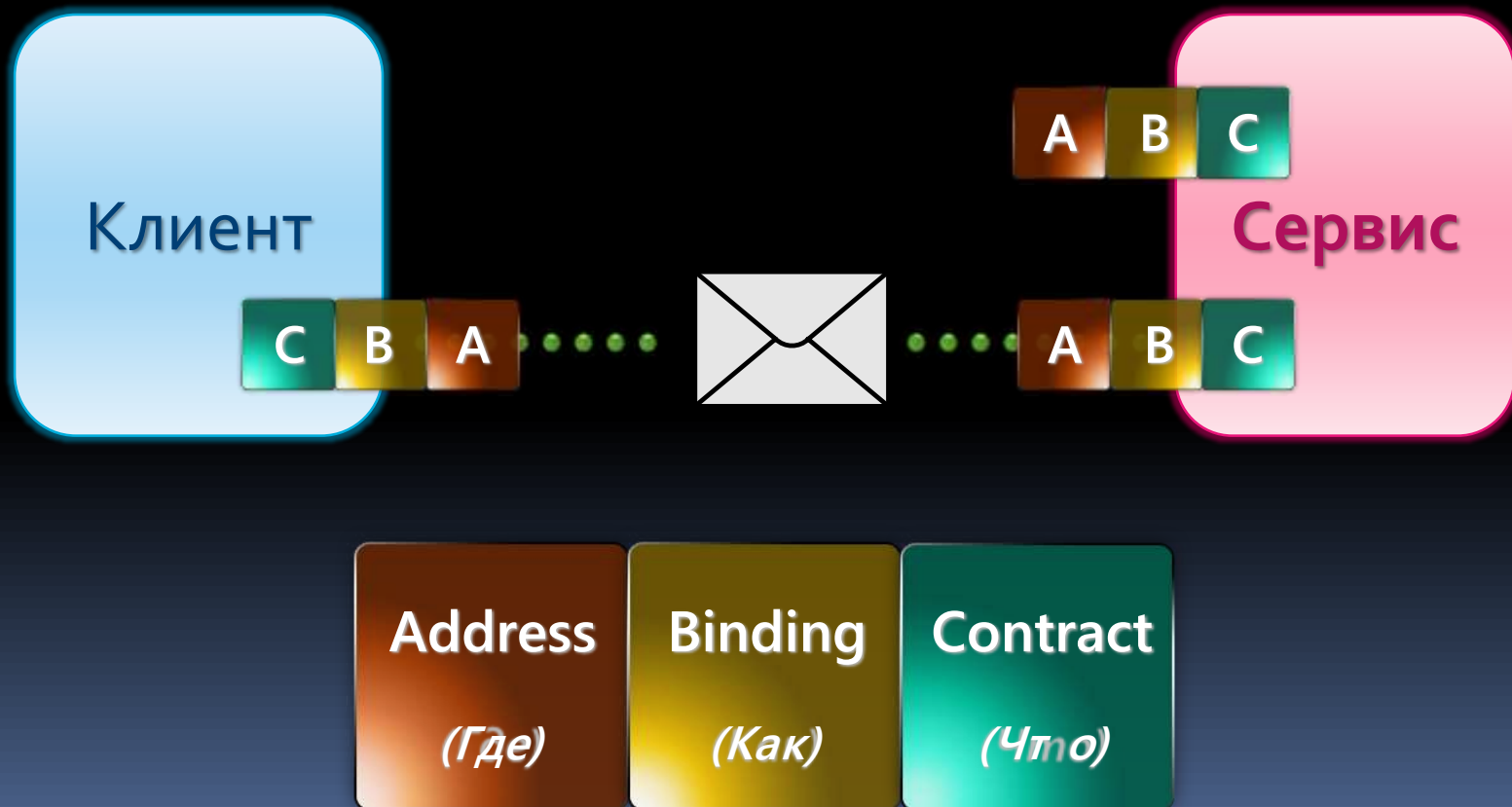




Лекция 25. Windows Communication Foundation

РАЗРАБОТКА ПРИЛОЖЕНИЙ НА ПЛАТФОРМЕ .NET

ABC в WCF



А

Адрес

- Задается типом `System.Uri` или в `*.config` файле
- Зависит от выбранной привязки
- В общем случае должны быть заданы
 - `scheme://MachineName[:port]/Path`
 - Схема. Транспортный протокол. HTTP, TCP и т.д.
 - Имя машины. DNS имя или IP адрес или др. в зависимость от схемы
 - Порт. Номер порта. Некоторые протоколы имеют порт по умолчанию и он может быть опущен
 - Путь. Путь к службе WCF
- Примеры
 - <http://localhost:8080/MyWCFService>
 - <net.tcp://localhost:8080/MyWCFService>
 - <net.pipe://localhost/MyWCFService>
 - [net.msmq://localhost/private\\$/MyPrivateQuery](net.msmq://localhost/private$/MyPrivateQuery)

Привязка

- Описывает
 - Транспортный уровень. Протокол передачи данных (HTTP, MSMQ, именованные каналы, TCP)
 - Тип канала (однонаправленный, запрос-ответ, дуплексный)
 - Механизм кодирования (двоичный, XML, SOAP)
 - Поддерживаемые протоколы Web-служб, если разрешены (WS-Security, WS-Transaction, и т.д.)

Контракт

- Интерфейсы – контракты служб WCF
- Интерфейсы для работы с WCF помечаются атрибутом `[ServiceContract]`
- Каждый метод в интерфейсе помечается атрибутом `[OperationContract]`
- Классы, реализующий контракты служб – типы служб `[ServiceContract]`

```
public interface ICalculator
{
    [OperationContract]
    Result SolveProblem (ComplexProblem p);
}
```

Хостинг службы WCF

- Публикует службу WCF, организует взаимодействие.
- В роли хоста может выступать:
 - любой тип приложения (консольное, WinForms, WPF)
 - Служба Windows
 - IIS (Internet Information Service)

Хост службы WCF – приложение

Необходимы сборка `System.ServiceModel` и такое же пространство имен.

Хост службы WCF – всегда класс `ServiceHost`

```
ServiceHost host =
```

```
    new ServiceHost(typeof(MyWCFService));
```

```
host.Open();
```

```
.....
```

```
host.Close();
```

Построение клиента

- Необходимы
 - общий контракт
 - привязка
 - адрес
- Статический импорт метаданных
 - Add Service Reference в Visual Studio
 - Утилита SvcUtil
 - Настроить руками. Сложно, но возможно
- Динамический импорт метаданных
 - Add Service Reference в Visual Studio для запущенного сервиса

Использование на клиенте

- Создан класс прокси MyServiceClient
- Класс прокси можно использовать как обычный класс. Но выполняться будет на сервисе

```
MyServiceClient client = new MyServiceClient();  
    int x = client .MyMethod(4, 5);  
    ...  
client.Close();
```

КОНТРАКТЫ

Односторонний вызов

- `IsOneWay=true` – односторонний вызов
- Особенности:
 - Выстрелил и забыл
 - Клиент продолжает работать сразу после вызова метода
- Применение
 - Простое асинхронное взаимодействие
 - Выполнение длительных запросов


Односторонний вызов

```
[ServiceContract]
public interface ICalculator
{
    [OperationContract(IsOneWay=true)]
    void StoreProblem (ComplexProblem
p);
}
```

Асинхронный вызов

- Применение
 - Простое асинхронное взаимодействие
 - Выполнение длительных запросов
- При генерации прокси необходимо указать о необходимости генерации асинхронных методов

```
MyServiceClient s = new MyServiceClient();  
IAsyncResult iRes = s.BeginMyMethod(5, null, null);  
  
while (!iRes.IsCompleted) Thread.Sleep(100);  
  
string result = s.EndMyMethod(iRes);
```



Асинхронный вызов и односторонний вызов



ДЕМОНСТРАЦИЯ

Контракт данных

- Используется при передаче пользовательских типов
- Тип снабжается атрибутом `[DataContract]`
- Переменные (данные) снабжаются атрибутом `[DataMember]`

`[DataContract]`

```
public class ComplexNumber
{
    [DataMember]
    public double real;
    [DataMember]
    public double Imaginary;
    [DataMember]
    public double Abs {get; set;};
    public double Real {get {return real;} set{ real = value;}};
    public ComplexNumber(double r, double i)
    {
        this.Real = r;
        this.Imaginary = i;
    }
}
```

Передача Enum

- Тип снабжается атрибутом `[DataContract]`
- Каждое значение Перечисления снабжается атрибутом `[EnumMember]`

`[DataContract]`

```
public enum MyNumbers
{
    [EnumMember]
    Один,
    [EnumMember]
    Два,
    [EnumMember]
    Три
}
```




Контракт данных

ДЕМОНСТРАЦИЯ

Сообщения об ошибках

- [ServiceContract]
- public interface ICalculator
- {
- [OperationContract]
- [FaultContract(typeof(DivideByZeroException))]
- ComplexProblem SolveProblem (ComplexProblem p);
- }

- Реализация метода:

- try { return n1 / n2; }
- catch (DivideByZeroException e) {
- DivideByZeroException f =
- new DivideByZeroException ("Calc Failure");
- throw new FaultException<DivideByZeroException>(f);
- }

- На клиенте просто ловится это исключение:
- catch (FaultException<DivideByZeroException> ex){}



Fault Contract

ДЕМОНСТРАЦИЯ

Двусторонний контракт

- Двусторонний контракт
 - Сервис может вызывать клиента
 - Повышенные требования к каналу
- Привязки:
 - `wsDualHttpBinding`, `netTcpBinding`,
`netNamedPipesBinding`
- Расширяет традиционную RPC-схему взаимодействия

Применение

- Оповещение
 - Подписка на события со стороны клиента
- Длительные вычисления
 - Процент выполнения
 - Возвращение данных порциями
- POLL vs. PUSH
 - POLL – клиент периодически опрашивает сервер о событии
 - PUSH – сервер «пинает» клиента по событию

Описание интерфейса

```
[ServiceContract(CallbackContract=  
    typeof(ICalculatorResults))
```

```
public interface ICalculatorProblems  
{  
    [OperationContract(IsOneWay=true)]  
    void SolveProblem (ComplexProblem p);  
}
```

```
public interface ICalculatorResults  
{  
    [OperationContract(IsOneWay=true)]  
    void Results(ComplexProblem p);  
}
```

- Вызов клиента:

```
ICalculatorResults callerProxy
```

```
    = OperationContext.Current.GetCallbackChannel<ICalculatorResults>();
```

```
callerProxy.Results(new ComplexProblem);
```

- Клиент должен реализовывать этот же CallbackContract
ICalculatorResults



Двусторонний контракт

ДЕМОНСТРАЦИЯ

Потоковая передача

- Передает данные как поток, а не как объект
- Привязки:
 - `basicHttpBinding`, `netTcpBinding`, `namedPipesBinding`
- Применение:
 - Данных много (не помещаются в буфер)
 - Данные доступны не сразу (передача видео)
 - Передача в реальном времени

Потоковая передача в коде

- Метод должен возвращать Stream:
 - `Stream getFile(string name);`
- Streaming нужно включить в конфигурации
 - `<binding transferMode="Streamed" />`
- После этого с ним можно работать, как с обычным потоком



ХОСТИНГ





IIS

ДЕМОНСТРАЦИЯ



Windows Service

ДЕМОНСТРАЦИЯ