# **Esercitazione di Fine Settimana – Week 4**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Sarah |
|  |  | Cognome | Tarantino |
|  |  | Data | 06/08/2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile fornendo anche degli esempi.

1. Cos’è un Web Service?

  Un **Web service** (**servizio web**) è un sistema software progettato per supportare l’interoperabilità tra diversi elaboratori(computer) su una medesima “rete informatica oppure in un contesto o “Sistema distribuito”.

Esso è in grado di mettersi al servizio di un applicazione comunicando su di una medesima rete tramite il protocollo HTTP. Un Web service consente quindi alle applicazioni che vi si collegano di usufruire delle funzioni che mette a disposizione. Una comune applicazione però non può fare è **mettersi in comunicazione** con un altro software. Un Web service infatti comunica tramite protocolli e standard definiti “aperti” e quindi sempre a disposizione degli sviluppatori.

I Web service hanno un’altra caratteristica molto particolare ed utile al loro scopo: sono **auto-contenuti** ed **auto-descrittivi**.

Un Web service è in grado di offrire un’interfaccia software assieme alla descrizione delle sue caratteristiche, cioè è in grado di far sapere che funzioni mette a disposizione (senza bisogno di conoscerle a priori) e permette inoltre di capire come vanno utilizzate. Ciò significa che, con una semplice connessione, anche senza conoscerlo, possiamo stabilire le operazioni che fornisce e possiamo subito iniziare ad usarle perchè ogni operazione ha una sua descrizione comprendente i parametri che si aspetta di ricevere, quelli che restituirà ed il tipo di entrambi

Questa tecnologia si basa su due caratteristiche fondamentali:

* **Multipiattaforma**: client e server non devono avere le stesse configurazioni per comunicare tra loro. Il servizio web fornisce quindi un terreno comune.
* **Condivisione**: nella maggior parte dei casi, un servizio web non è disponibile solo per un unico client, anzi, spesso vi accedono molteplici client contemporaneamente tramite Internet.

Quando viene utilizzato un Web service, un client invia una richiesta a un server e vi provoca un’azione. Il server invia quindi una risposta al client.

Un servizio web viene indirizzato tramite un **Uniform Resource Identifier** (URI) unico. Simile a un Uniform Resource Locator (URL), che consente di indirizzare i siti web, l’URI è un indirizzo per il corrispondente Web service. Teoricamente gioca un ruolo anche il servizio di directory UDDI, attraverso il quale si possono trovare i servizi web. Questo servizio, tuttavia, non ha mai preso piede e i maggiori sostenitori si sono ritirati dal progetto.

I servizi web dispongono di un**'interfaccia** (descritta in un formato automaticamente elaborabile sotto forma di **file scritto in linguaggio WSDL(Web Service Description Language)** che descrive il servizio in modo più dettagliato. Essa **espone all'esterno** il servizio/i associato/i e utilizzandola altri sistemi possono interagire con l'applicazione stessa attivando le operazioni descritte in essa (servizi o richieste di procedure remote) tramite appositi "messaggi" di richiesta: tali messaggi di richiesta sono inclusi in una "busta".

Infine, la **comunicazione** funziona **attraverso diversi protocolli e architetture**. Un protocollo molto diffuso è, ad esempio, *SOAP*, che viene combinato con lo *standard Internet HTTP o RESTful Web services*. Queste tecniche vengono utilizzate per inviare e ricevere richieste e risposte spesso formattate secondo lo standard XML.

La comunicazione si basa spesso *sul linguaggio* ***XML*** (Extensible Markup Language). Questo linguaggio relativamente semplice può essere compreso sia dalle persone che dai computer ed è ideale per collegare sistemi con prerequisiti diversi. *REST* invece è compatibile anche con altri formati, come ***JSON***.

Innanzitutto possiamo dire che sono coivolte due parti il **Client** che si preoccupa di inviare una richiesta sotto forma di messaggio e che comporta un azione) tramite l’utilizzo di un chiamata( una funzione, subroutine o metodo), ed il **Server** che riceve la chiamata, formula una risposta(Messaggio, e la invia a. Client.

Per descrivere un esempio di servizio web potremmo partire da un software scritto in java/c# visula basic ... che gira su un dispositivo Windows. **Il programma richiede il servizio** di un **server** web Apache. Per farlo, **il client invia una richiesta** SOAP/REST al server sotto **forma di messaggio HTTP**. *Il servizio web interpreta il contenuto* della richiesta **e fa in modo che il servizio esegua un’azione**. Dopo l’esecuzione**, il servizio web formula una risposta che rimanda al client**, sempre con SOAP/REST e HTTP. Qui la **risposta viene interpretata nuovamente** e l’informazione viene **trasferita al software** dove viene ulteriormente elaborata.

1. Quali parti costituiscono una HTTP Response?

Due parti:

* Headers (contiene lo Status Code)
* Body

1. Descrivere l’utilizzo degli attributi [DataContract] e [OperationContract]

Il [DataContract] viene utilizzato nel progetto Core nelle classi che solitamente si trovano nella Folder Model, per identificare le che descrivono il DataModel di riferimento, ogni Attributo che si vuole conivolgere nelle operazioni coinvolte nelle chiamate viene marcato come [DataMember].

[OperationContract] è viene utiizzato per definire i metodi dell’Interfacca che definisce il Servizio in un progetto Wcf

1. Per cosa vengono utilizzati gli HTTP Methods /Verbs in un servizio REST?
2. Vengono utilizzati nelle annotazioni che precedono l’implementazione dei metodi delle operazione CRUD(GET, POST(CREATE),PUT(UPDATE), DELETE) nella classe Controller del progetto WebApi
3. Vengono utilizzati per effettuare le chiamate vere e propriere nel Client ovvero nella Classe programm del progetto ApiClient
4. Come viene configurato un servizio realizzato con ASP.NET Core WebAPI?

Viene Configurato nel file Startup.Cs, nel metodo **IConfigurationServices**

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

services.AddControllers();

// services.AddTransient<IMyService,Myservice()>

services.AddScoped<MainBusinessLayer>();

services.AddTransient<IEmployeeRepository, EFEmployeeRepository>();

// services.AddSingleton<IMyService,MyService()>

services.AddDbContext<EmployeeContext>(options =>

{

options.UseSqlServer(Configuration.GetConnectionString("employeeDb"));

});

**Esercitazione Pratica**

* Realizzare un database per la Gestione degli Ordini e dei Clienti.
  + ***Cliente***
    - *ID* (int, PK), *CodiceCliente* (string), *Nome* (string), Cognome (string)
  + ***Ordine***
    - *ID* (int, PK), *DataOrdine* (date), *CodiceOrdine* (string), *CodiceProdotto* (string), *Importo* (decimal)
    - Ogni Ordine è legato ad un Cliente
  + La realizzazione dello strato di accesso al dato deve essere realizzata con EF Code-first, applicando il Repository Pattern
* Realizzare un servizio WCF per la gestione di una Anagrafica Clienti (CRUD)
* Realizzare un servizio REST per la gestione di una Anagrafica Ordini (CRUD)
* Realizzare un client (Console app) per:
  + CRUD Clienti
  + CRUD Ordini
  + Stampa Elenco Ordini per uno specifico Cliente
  + Stampa Dettagli Ordine per uno specifico Ordine
  + Report Ordini per Anno
    - Per ogni anno specificare numero di ordini e importo totale ordini