Universidade de Brasília - Campus Gama
Orientação a Objetos
Exercício de Programação 2

Atenção: Este é um exercício de programação que visa a aplicação de alguns conceitos de orientação a objetos aprendidos durante a disciplina. Recomenda-se o trabalho contínuo em tal exercício, pois o mesmo dificilmente será implementado em um ou dois dias com a devida qualidade exigida. Este exercício busca avaliar os seguintes conceitos:

- Controle de versão do código fonte
- Criação de classes e objetos.
- Utilização de atributos, métodos e construtores.
- Utilização de herança.
- Utilização de interfaces gráficas.
- Teste unitário.

Para o melhor aproveitamento do aluno, recomenda-se que o mesmo busque várias referências externas para auxiliar na resolução do exercício.

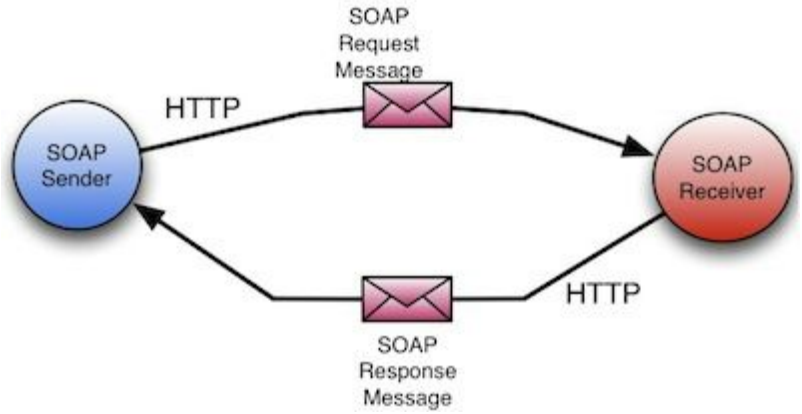
O objetivo deste trabalho é utilização de interfaces **swing** para a criação de uma aplicação gráfica que seja capaz de acessar uma base de dados externos e exibí-los de forma organizada para o usuário, fornecendo a possibilidade de filtros e busca.

A conexão com a base de dados bem como o tratamento dos dados recebidos serão fornecidos através de métodos já implementados, devendo todas as demais funcionalidades serem desenvolvidas pelo aluno.

1. Fundamentação Teórica

1.1. Webservice

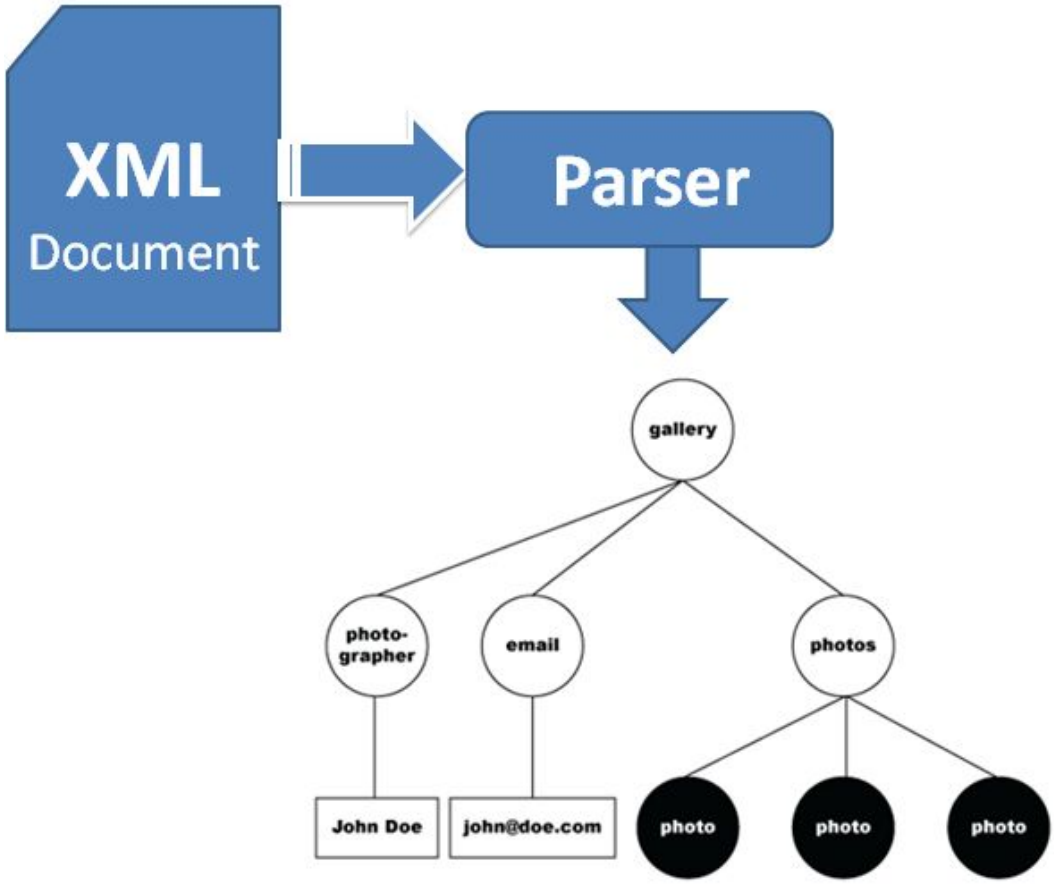
Webservice é um serviço web que permite o consumo de um conteúdo de uma aplicação *web* por outra aplicação através de um mecanismo de requisição e resposta. *Webservicces* são fundamentais para a transmissão de informações entre aplicações externas. Um dos protocolos mais utilizados de webservice é o protocolo SOAP (*Simple Object Access Protocol* - Protocolo Simples de Acesso a Objetos) que se baseia na linguagem XML (*eXtensible Markup Language* - Linguagem e Marcação Extensível).



Comunicação via protocolo SOAP

1.2. Parser

Um *parser* é um programa que transforma uma fonte de entrada de dados em partes menores para que sejam melhor gerenciadas por outras aplicações. Neste contexto, o *parser* será utilizado para transformar o arquivo XML recebido do servidor SOAP em objetos e seus atríbutos para que possam ser manipulados dentro da aplicação Java.



2. Exercício

1.1. Objetivos

O objetivo deste trabalho é consultar o *WebService* da Câmara dos Deputados à fim de obter as informações dos deputados federais em exercício e exibí-los através de uma *interface* gráfica agradável e informativa, com possibilidade de filtragem e pesquisa.

Para auxiliar no trabalho, **serão disponibilizados**:

- O *parser* que realizará a comunicação com o *WebService* e transformará os dados recebidos em XML para objetos.
- As classes principais da aplicação: Câmara, Comissão, Deputado, Detalhe (do deputado) e Partido.

Através do Parser, serão disponibilizadas as seguintes informações sobre os deputados:

Nome	Tipo	Descrição
ideCadastro	Integer	ID do parlamentar
condicao	String	Informa se o deputado é titular ou suplente
nome	String	Nome civil do parlamentar
nomeParlamentar	String	Nome de tratamento do parlamentar
sexo	String	Sexo (masculino ou feminino)
uf	String	Unidade da Federação de representação do parlamentar
partido	String	Sigla do partido que o parlamentar representa
gabinete	String	Numero do Gabinete do parlamentar
anexo	String	Anexo (prédio) onde o gabinete está localizado
telefone	String	Numero do telefone do gabinete
email	String	Email institucional do parlamentar

1.2. Fase Zero: Preparação

Visando manter seu projeto organizado, deve-se utilizar o seguinte padrão de organização:

- **build**: Pasta onde serão gerados os arquivos compilados.
- **src:** Pasta que manterá todo o código fonte
  - **controllers**: Pasta que conterá as controladoras das models

- **models:** Pasta que conterá as models do projeto
- **views:** Pasta que conterá as views do projeto

- **doc**: Pasta onde conterá qualquer informação extra de documentação.

Por fim, crie os seguintes arquivos:

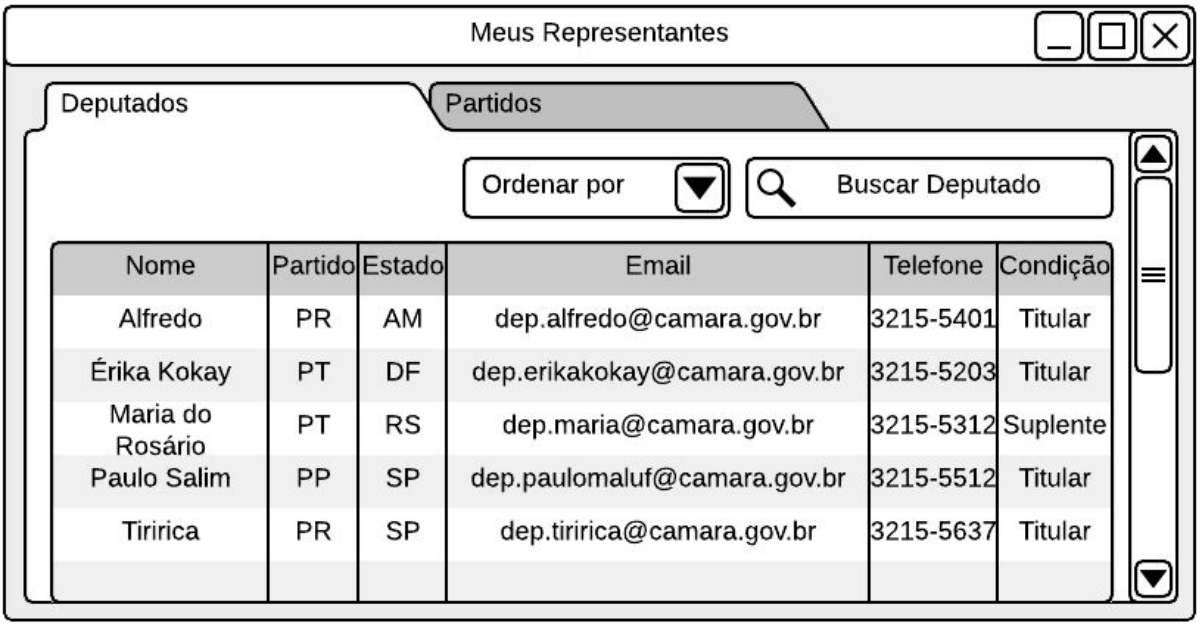
- **README.md**: Explique de forma ORGANIZADA e direta como compilar e executar o seu programa.

1.3. Fase Um: Interface

A interface deverá ser construída utilizando o *toolkit* SWING, que já é parte do próprio Java. Algumas IDEs possuem editores visuais de interfaces, facilitando o design e a edição de componentes visuais. Alguns exemplos são Netbeans, Eclipse e Intellij IDEA.

O *layout* das telas é livre, cada aluno poderá desenhar da maneira como preferir. Todavia, os seguintes itens deverão estar presentes:

- Uma lista com todos os deputados;
- Deputados por partido;
- Controles para ordenação da lista de deputados com pelo menos dois parâmetros diferentes;
- Contagem de deputados em cada partido;
- Visualização detalhada de um deputado selecionado numa nova janela da aplicação, exibindo todas as informações disponíveis do deputado;



Exemplo de *interface* da aplicação

1.4. Fase Dois: Testes unitários

Todo o código produzido, com exceção das *views*, deve possuir testes unitários para garantir que todas as funcionalidades da aplicação funcionam perfeitamente. Os testes também devem estar em umas pasta separada, denomina **tests,** contendo as pastas:

- **models:** teste unitário das *models.*
- **controllers:** teste unitário das *controlles.*

8. Critérios de Avaliação

Critério	**Descrição**	**Peso**
Interface	Disposição das informações; Facilidade de visualização dos dados;	3
Funcionalidades	Utilização de filtos; Variedade de pesquisas disponíveis; Exibição dos detalhes dos deputados;	3
Testes	Testes unitários das models, e controllers.	2
Organização e Arquitetura	Utilização adequada da arquitetura MVC (*Models* - *Views* - *Controllers*); Organização do código.	2
Extra	Exibição das fotos do deputado (0,25); Interface gráfica e interação do usuário com a aplicação acima do esperado (0,25); Melhoria do *parser* para a obtenção das proposições dos deputados (0,5).	1