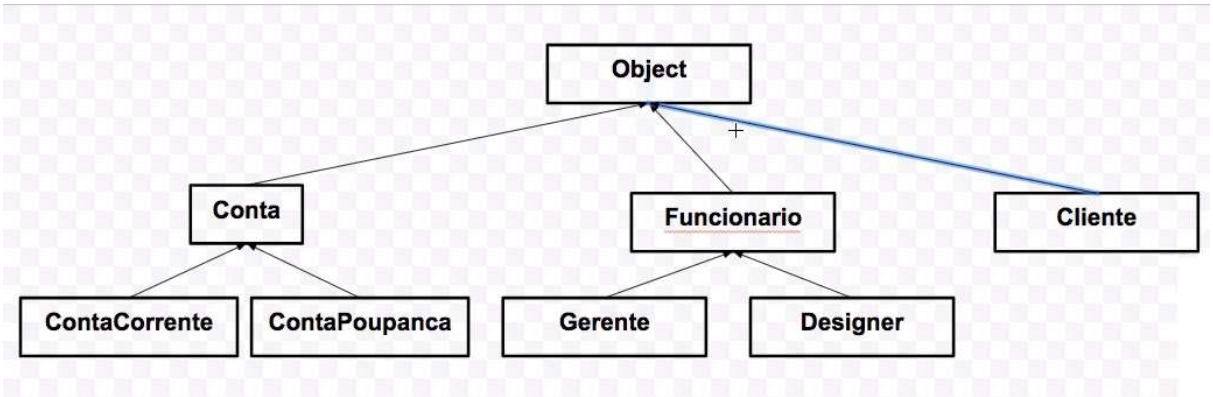
Java.lang: é um pacote que inclui classes essenciais para qualquer programa. É o único pacote que não necessita ser importado. Por ser de suma importância para o desenvolvimento de qualquer aplicação Java, ele é incluído automaticamente. Pertencem a esse pacote a classe String (FQN: java.lang.String), System, Exception etc.

- Classe String: Sendo String uma classe, para facilitar a vida do desenvolvedor, não precisamos utilizar o new todas as vezes que formos trabalhar com Strings, mas nada impede que você o faça. As duas formas são funcionais, embora o segundo caso seja considerado uma má prática. A partir da primeira forma a máquina virtual consegue executar algumas otimizações, o que é impossível no segundo caso (com new).
  - Uma vez criada a String, ela não poderá ser modificada posteriormente (imutabilidade). Caso você queira alterar algo em uma String, você terá de criar uma String que refletirá uma nova ação, ou seja, teremos dois objetos. Todos os métodos (por exemplo, o replace) funcionam nessa linha: devolvem uma nova String, respeitando o conceito de imutabilidade.
  - A classe String é uma CharSequence (O tipo CharSequence é uma interface que a própria classe String implementa) se precisamos concatenar muitos String devemos usar a classe StringBuilder.

- Classe System: System.out.println("Alura");
  - System: é uma classe pública e provém do pacote java.lang.

- ➢ out: é um atributo público, pois é acessado fora da classe. É também uma referência estática (static) ou seja, precisa acessar a classe System e não uma referência desta.
- ➢ println(): é um método. Como o elemento anterior ao println() é uma referência, o método não possui acesso estático. Há algumas outras informações acerca de println(): ele pode receber uma String, mas também um int. Aparentemente, existem várias versões de um mesmo método, ou seja, muitas sobrecargas. O println() não joga exceções do tipo checked, nunca fomos obrigados a fazer algum tratamento de exceção.

❖ Object: é a classe raiz (mãe) do Java. Qualquer objeto pode ser referenciado pelo tipo Object, já que ela é a principal forma mais genérica de referenciar um objeto. Não é preciso deixar explícito na declaração de uma classe que ela deve herdar de Object, porque isso é automático.



- ➢ toString(): é um método da classe Object. Devolve informações sobre o estado do objeto. É útil para a depuração no desenvolvimento. Esse método existe para

ser sobrescrito e assim dar um significado maior do que a saída padrão desse método.

- ➢ equal(): é um dos métodos fundamentais da classe Object. Utilizado pelas listas, Para usar, devemos sobrescrever o método para definir a igualdade do objeto.

❖ Classe Wrapper: No Java, há uma classe para cada tipo primitivo. Essas classes se chamam wrappers. Isso porque elas "embrulham" o tipo primitivo do objeto, que internamente guarda o valor primitivo. Elas existem para que haja compatibilidade com as coleções, nos permitindo, por exemplo, guardar números dentro de uma lista.

Tamanho	Tipo primitivo	Wrappers
8 bytes	double	`java.lang.Double`
4 bytes	float	`java.lang.Float`
8 bytes	long	`java.lang.Long`
4 bytes	int	`java.lang.Integer`
2 bytes	short	`java.lang.Short`
1 bytes	byte	`java.lang.Byte`
2 bytes	char	`java.lang.Character`
	boolean	`java.lang.Boolean`

Sendo que, double e float são flutuantes, long, int, short e byte são inteiros, char representa um caractere, e por fim, temos um booleano.

A transformação do tipo primitivo para o objeto referência acontece automaticamente, e é chamada de autoboxing. O caminho inverso é chamado de unboxing.

- ➢ Parsing: Transformação de um tipo para outro.
- ➢ Atenção: A existência de primitivos e wrappers é explicada pelo momento da criação do Java, à época, a capacidade de processamento das máquinas era limitada, e a memória era custosa, portanto, pensando em questões de desempenho, e memória, era importante a existência dos primitivos. Eles são mais rápidos, e ocupam menos espaço.

❖ Number: Classe mãe que conecta todas as classes wrappers que representam um valor numérico. A classe Number é estendidas pelas seguintes classes: Double, Float, ambas flutuantes, Long, Integer, Short e Byte, representando os inteiros.