 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

## Objetivos

Com a realização do trabalho prático, pretende-se que os alunos ponham em prática todos os conhecimentos adquiridos na utilização do paradigma de programação orientado a objetos (POO) e a sua implementação na linguagem de programação Java, demonstrando as suas apetências em:

- Conhecer e compreender os conceitos fundamentais associados ao paradigma da programação orientada a objetos;
- Conceber e implementar, para problemas concretos, soluções que tenham por base o paradigma da programação orientada a objetos.
- Reconhecer e compreender a semântica e a sintaxe da linguagem Java.
- Reutilizar, alterar e desenvolver código recorrendo à linguagem Java tendo em vista um determinado problema com regras semânticas específicas.

Considere ainda que:

- Não é permitida a utilização de API's/conceitos Java que não tenham sido alvo de lecionação no ano letivo corrente da unidade curricular Paradigmas de Programação. Os alunos que pretendam utilizar API's adicionais devem atempadamente pedir autorização a um dos docentes da unidade curricular.
- Não é permitida a utilização de coleções Java predefinidas ([Java Collections Framework](#)).
- Os recursos de suporte ao trabalho referenciados no enunciado, são de utilização obrigatória.

## Enunciado

A empresa *PackSolutions* tem como principal atividade a disponibilização de soluções dedicadas ao empacotamento de encomendas de grande dimensão das mais diversas empresas da região.

Como resultado de um processo de modernização tecnológica, a empresa pretende desenvolver uma API em linguagem Java capaz de suportar os requisitos de uma ferramenta de suporte à gestão de encomendas e à acomodação de caixas (**Box**) dentro dos contentores (**Container**) existentes em cada encomenda. Desta forma, os colaboradores da *PackSolutions* podem mais facilmente gerir e visualizar os itens expedidos para cada encomenda.

## Requisitos gerais

A solução a desenvolver deverá ter capaz de criar e gerir encomendas (**Order**) para envio (**Shipping**). Cada encomenda tem pelo menos um contentor (**Container**). Um **Container** representa uma “caixa” de grandes dimensões especialmente orientada para acomodar um conjunto de **Items** (caixas/**Box**) de menor dimensão que armazenam os itens a transportar, como ilustrado na Figura 1.

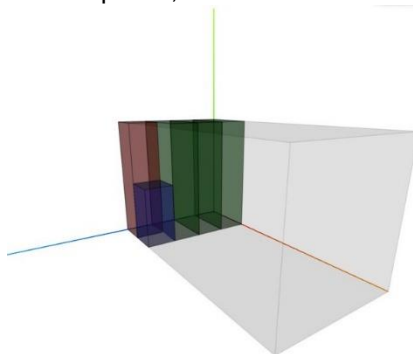



Figura 1. Exemplo de um container de uma encomenda (a cinzento o **Container** e os **Items** internamente representados em diversas cores)

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

A utilização da API deverá contemplar a especificação das dimensões do **Container** e tendo por base essas dimensões, deverá definir os **Items** que pretende acomodar e em que posições (considerando as coordenadas cartesianas).

Considere que:

- A API deverá suportar a gestão de encomendas, sendo que inicialmente, uma encomenda apenas possui os **items** que têm de ser transportados e **obrigatoriamente** obtidos através da sua importação através de um ficheiro JSON (ver detalhes na secção: Ficheiros JSON envolvidos no desenvolvimento da API);
- Cada encomenda poderá ser enviada por diversos **shippings**;
- Quando os detalhes de envio são definidos (**shippings**), os **items** são atribuídos a **containers** específicos.;
- Os itens só podem ser enviados uma vez. Ou seja, não podem ser posicionados em múltiplos **containers**;
- A encomenda é considerada “fechada” quando todos os seus itens são recebidos pelo destinatário.
- É necessário garantir restrições específicas relacionadas com o envio dos itens da encomenda (ver documentação *javadoc*).
- A componente de gestão de encomendas (**Management**) é **fundamental** (requisito obrigatório) para a conclusão da API, uma vez que sem essa gestão, não é possível suportar os requisitos operacionais.
- A divisão de encomendas por múltiplos **shippings** é **fundamental** (requisito obrigatório) para a operacionalização das encomendas.

O objetivo passa por facilitar o processo de acomodação de itens em contentores, otimizando os custos de expedição das encomendas. Note que a API deverá apenas suportar o posicionamento de caixas de forma manual (com coordenadas cartesianas especificadas pelo utilizador, ver Figura 2).

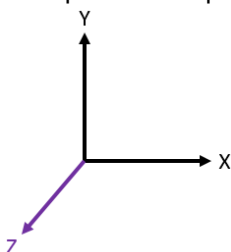


Figura 2. Eixos


## Descrição Técnica

Como suporte ao desenvolvimento da API, são disponibilizados um conjunto de recursos (*ER\_Resources*), de utilização obrigatória e que definem os contratos que permitem o desenvolvimento da API. Os conteúdos fornecidos são um complemento ao presente enunciado, contendo informação específica sobre as particularidades de implementação de cada funcionalidade. A utilização dos contratos constitui um ponto de partida, cujos ficheiros não podem ser alterados. **Caso não utilize ou altere os recursos disponibilizados, todo o trabalho é invalidado.**

Deverá realizar a implementação do código necessário para suportar cada uma das operações definidas nos contratos. Teste o mais exaustivamente possível o código que desenvolveu como resposta aos requisitos apresentados. Recorra a comentários JavaDoc e não só de modo a documentar, o mais exaustivamente possível, o código que desenvolveu.

A existência dos contratos não deve ser impeditiva para a implementação de novas funcionalidades e/ou novos métodos ou classes.

## Interface gráfica

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

Para complementar o processo de validação da API a desenvolver, é disponibilizado um componente que com base no código desenvolvido, apresenta uma *interface* gráfica. Poderá utilizar esta componente para **validação do código** produzido. A componente de interface gráfica **já se encontra implementada**, devendo ser utilizada após a implementação e validação da API.

A interface gráfica é exposta através da classe: `PackingGUI` que disponibiliza um método `render` responsável por apresentar uma interface gráfica, utilizando o `browser` definido por defeito. No contexto da API, os ficheiros JSON<sup>1</sup> suportam o armazenamento dos dados de uma encomenda e ainda a exportação de dados de encomendas de forma a que possam ser apresentados através de visualizações web específicas. O JSON é um formato compacto para troca de dados entre sistemas, que suporta o armazenamento de dados no formato atributo-valor (ver <https://www.json.org/>).

Existem duas possibilidades de invocação do método `render`:

- `PackingGUI.render(String pathToFile)`: Como parâmetro, o método `render` recebe uma `string` com o caminho para um ficheiro JSON que representa uma encomenda (**Order**). Através de um exportador (`IExporter`), deverá exportar uma dada encomenda para o formato JSON apresentado na Figura 3. Pode utilizar o método `validate` para validar o ficheiro JSON utilizado para a renderização.
- `PackingGUI.render(String[] pathToFiles)`: Como parâmetro, o método `render` recebe uma coleção de `strings` que representam caminhos para ficheiros JSON que armazenam uma representação compatível com as visualizações disponibilizadas pelo website: <https://quickchart.io/>. Cada visualização está associada a uma estrutura JSON que é utilizada para enviar os dados que serão apresentados de acordo com as características da visualização. Existem diversas visualizações que pode tirar partido para a apresentação de resultados. Através de um exportador (`IExporter`) deverá exportar **três** visualizações:
  - Para as encomendas em que existem itens por enviar, apresentar a percentagem de itens por enviar.
  - Apresentar a percentagem de encomendas em cada estado.
  - Visão geral dos clientes e o respetivo número de encomendas.
- **Deverá implementar a lógica necessária para suportar as visualizações.**

## Ficheiros JSON envolvidos no desenvolvimento da API

A implementação da API requer três exportações distintas para ficheiros JSON:

1. Exportação de um **shipping** de uma encomenda para visualização 3D
2. Exportação de dados estatísticos através de visualizações distintas;
3. Exportação de uma folha de monitorização para uma encomenda, onde são exportadas informações essenciais relacionadas com as encomendas (incluindo os dados gerais da encomenda e dos seus **shippings**). A estrutura do ficheiro exportado (*schema*) fica ao critério de cada grupo.

O ficheiro JSON relacionado com **a exportação de um shipping de uma encomenda** considera:

- Os dados gerais da encomenda: identificador (`id`), data (`date`), estado (`status`), dados do cliente com respetivas moradas (`customer`) e da pessoa de destino com respetiva morada (`destination`) para entrega da encomenda.
- Os dados de cada container (armazenados num array), contendo a informação do contentor: referência (`reference`), o volume máximo e ocupado (`volume` e `occupiedVolume`), as dimensões (`length`, `height`, `depth`) e a cor do contentor e das bordas (`color` e `colorEdge`).
- Os dados de cada item (armazenados num array) dentro de um contentor, contendo a informação de cada item: a referência (`reference`), descrição (`description`), as dimensões (`length`, `height`, `depth`), cor do item e das bordas (`color` e `colorEdge`) e as coordenadas cartesianas do item no contentor (`x`, `y` e `z`).

<sup>1</sup> JavaScript Object Notation

<div>P.PORTO</div> <div>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</div>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

```
{
  "id": 1,
  "date": {
    "day": 1,
    "month": 1,
    "year": 2020
  },
  "status": "RECEIVED",
  "customer": {
    "id": 1,
    "name": "John Doe",
    "address": {
      (...)
    },
  },
  "destination": {
    "name": "Jane Doe",
    "address": {
      "country": "country A2",
      "number": 2,
      "street": "street A2",
      "city": "city A2",
      "state": "state A2"
    },
  },
  "billingAddress": {
    (...)
  },
  "containers": [
    {
      "reference": "c1",
      "volume": 125000,
      "occupiedVolume": 36,
      "depth": 50,
      "length": 50,
      "height": 50,
      "color": "white",
      "colorEdge": "black",
      "closed": true,
      "items": [
        {
          "reference": "ITEM1",
          "description": "ITEM1",
          "depth": 1,
          "length": 1,
          "height": 1,
          "color": "aqua",
          "colorEdge": "aqua",
          "x": 0,
          "y": 0,
          "z": 0,
        },
        (...)
      ],
    },
    (...)
  ],
}
```

Figura 3. Excerto de um documento JSON com dados de uma encomenda

Os ficheiros JSON relacionados com a geração de **visualizações** com dados de encomendas, devem ser gerados de acordo com as várias visualizações disponibilizadas em <https://quickchart.io/> (neste website, encontra sinalizado a amarelo, a estrutura do documento JSON correspondente a cada visualização). Por exemplo, para criar um gráfico de barras, deve ser gerado um documento JSON de acordo com o exemplo apresentado na Figura 4. Para gerar um gráfico circular, deve ser gerado um documento JSON semelhante ao apresentado na Figura 5.

<div>P.PORTO</div> <div>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</div>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

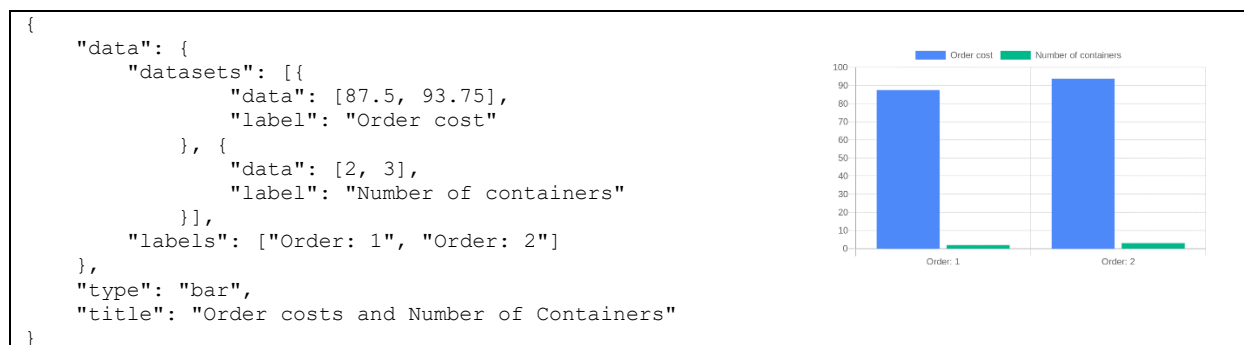


Figura 4. Documento JSON para apresentação de um gráfico de barras (ilustrado à direita).

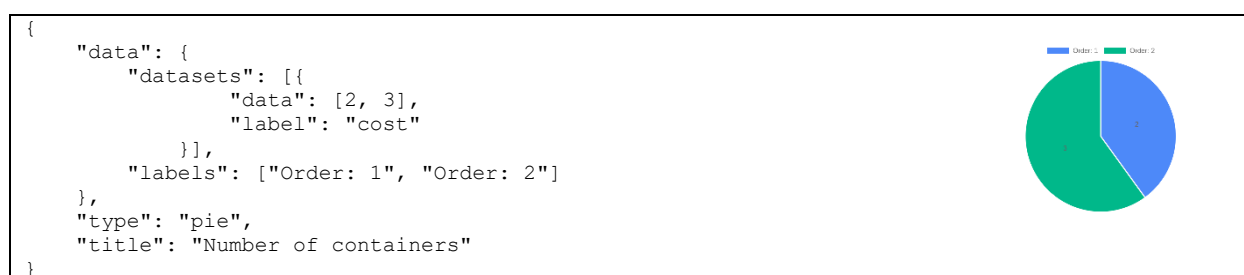



Figura 5. Documento JSON para apresentação de um gráfico circular (ilustrado à direita).

Também deverá ser possível **importar** encomendas a partir de um documento JSON. O documento de leitura deverá apenas conter os dados genéricos da encomenda, assim como os itens que serão posteriormente empacotados numa encomenda para envio (Figura 5). Considere parte da descrição apresentada para a Figura 3.

```
{
  "id": 1,
  "date": {
    "day": 1,
    "month": 1,
    "year": 2020
  },
  "customer": {
    "name": "John Doe",
    "vat": "111111",
    "address": {
      "country": "country A1",
      "number": 1,
      "street": "street A1",
      "city": "city A1",
      "state": "state A1"
    },
    "billingAddress": {
      "country": "country A1",
      "number": 1,
      "street": "street A1",
      "city": "city A1",
      "state": "state A1"
    }
  },
  "destination": {
    "name": "Jane Doe",
    "address": {
      "country": "country A2",
      "number": 2,
      "street": "street A2",
      "city": "city A2",
      "state": "state A2"
    }
  }
}
```

 <small>ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO</small>	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

```

    "items": [
      {
        "reference": "ITEM1",
        "depth": 1,
        "length": 1,
        "height": 1,
        "description": "desc 1"
      }, (...)
    ]
  }

```

Pode utilizar uma biblioteca Java para facilitar a leitura/escrita do documento. A escolha da biblioteca de manipulação de documentos JSON a utilizar fica ao critério de cada grupo. A título de exemplo, é disponibilizado na plataforma moodle um excerto de código demonstrativo de utilização da biblioteca: json-simple<sup>2</sup>).

#### Nota:

A interface gráfica deve apenas ser utilizada após a implementação e validação de todo o código desenvolvido. Isto significa que tem de realizar a instanciação das diversas classes de forma a proporcionar a exportação (através da interface *IExporter*) do documento JSON que é utilizado pela interface gráfica.

## Elaboração do trabalho

Este trabalho é realizado em grupo que deverá ser composto no **por 2 alunos** da unidade curricular. Os alunos devem comunicar atempadamente o seu grupo de trabalho na plataforma moodle, até ao dia **10 de julho 2020**.

### Datas e considerações

O trabalho deve ser entregue até às **23:55** horas do dia **13 de julho de 2020**, devendo a entrega ser feita através da página da unidade curricular de Paradigmas de Programação em <http://moodle.estg.ipp.pt>.

A defesa será realizada por turnos no dia do exame de recurso (a partir das 14:30) e o horário exato para cada aluno será comunicado na plataforma moodle após a entrega do trabalho.


Considera-se por defesa satisfatória, quando o aluno demonstra que realizou o trabalho submetido e que **domina todos os conceitos de programação orientada a objetos aplicados na resolução do trabalho**. Tentativas de fraude, resultarão na avaliação do trabalho como: **Fraude Académica**.

### Formato da entrega

Os trabalhos entregues deverão evitar (se possível) utilizar caminhos absolutos ou endereços específicos, de modo a que possam ser facilmente utilizados em qualquer máquina. Para além disso, e no sentido de facilitar a receção dos vários trabalhos recebidos, estes deverão observar as seguintes regras:

- Todos os elementos do grupo deverão submeter o trabalho no link respetivo (Entrega do Trabalho);
- O trabalho desenvolvido deverá ser entregue através do moodle, através da submissão de um ficheiro com o nome PP\_EN\_<nr\_do\_aluno>\_<nr\_do\_aluno>.zip, contendo:
  - Os ficheiros criados incluindo o(s) projeto(s) do IDE Netbeans e uma pasta com a distribuição (jar) da solução proposta.
  - Recorra a comentários JavaDoc, e não só, de modo a documentar, o mais exaustivamente possível, o código desenvolvido.
  - Cada ficheiro de código entregue por cada grupo terá de possuir no início do mesmo um comentário com pelo menos a seguinte informação (com as adaptações óbvias para cada aluno/grupo):

<sup>2</sup> <https://code.google.com/archive/p/json-simple/>

 ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO	Tipo de Prova Época Recurso	Ano letivo 2019/2020	Data 14/07/2020
	Curso LEI/LSIRC	Hora 14:30	
	Unidade Curricular Paradigmas de Programação	Duração	

/\*

\* Nome: <Nome completo do aluno>

\* Número: <Número mecanográfico do aluno>

\* Turma: <Turma do aluno>

\*

\* Nome: <Nome completo do colega de grupo>

\* Número: <Número mecanográfico do colega de grupo>

\* Turma: <Turma do colega de grupo>

\*/

Os alunos que não realizem a entrega do trabalho até à data/hora definida serão sujeitos a **penalização** ou a **invalidação do trabalho**.