

装满满智能装箱API服务

市场版本: v2.0

更新日期: 2022-03-31

装满满智能装箱API服务

服务简介

结果可视化

调用方式

请求参数

Body参数数据结构

Body参数示例

返回体报文数据结构

返回成功示例

调用失败示例

参数含义介绍

参数引用关系

坐标系

货物尺寸与旋转方向

指定容器

容器尺寸

容器数量

算法配置

装载结果解读

服务声明

服务简介

本产品基于RESTful API模式，使用HTTP调用方式运行。请用户仔细阅读调用方法，按照指定格式发送输入数据，获取装箱结果。

请注意

1. 此版本API主要针对单一柜型的简单装载计算，用户在每次调用时需分别输入待装货物和备选容器的规格、数量信息，API将返回不超过用户指定容器数的装载结果，并返回可能的剩余货物，详细信息请参见使用说明文档
2. 建议使用毫米作为长度单位，并使用**整型**表示；建议使用千克作为重量单位，可接受**浮点数**
3. 此版本API对输入货物的规格信息和数量实行分开管理，其中：
 1. "items" 字段包含所有待装货物的属性信息，每种货物以 "value::uid" 进行区分，uid原则上可以是任意内容的字符串，请保持每种货物的uid唯一；
 2. "inventory_items" 字段表示待装货物的数量清单，每种货物以 {"uid": "quantity"} 的方式指定具体的数量，其中 "uid" 表示具体是哪种货物，quantity 表示该种货物的数量，算法将根据货物uid在 "items" 字段中查询货物规格详情，请保证在这里输入正确的货物uid
4. 此版本API对输入容器的规格信息和数量实行分开管理，其中：
 1. "containers" 字段包含所有备选容器的属性信息，每种容器以 "value::uid" 进行区分，uid原则上可以是任意内容的字符串，请保持每种容器的uid唯一；

2. "inventory_containers" 字段表示备选容器的数量清单，每种容器以 {"uid": "quantity"} 的方式指定具体的数量，其中 "uid" 表示具体是哪种容器，quantity 表示该种容器可以使用的数量，算法将根据容器uid在 "containers" 字段中查询容器规格详情，请保证在这里输入正确的容器uid
5. 货物允许朝向使用数字编号[0,1,2,3,4,5]表示，最多6种朝向；其中数字0表示输入长宽高的原始朝向，其余1~5表示货物经过旋转后的朝向，详细信息请参见使用说明文档
6. "configuration" 字段中包含可接受的算法参数，默认提供以下功能，详细信息请参见使用说明文档：
 1. 限定容器的载重上限
 2. 配置轻重货的堆叠关系
 3. 配置大小货的堆叠关系
 4. 配置货物地面悬空比例

结果可视化

用户可将完整的返回报文存储为本地JSON文件，再上传至<https://aparecium.doraclp.cn/index.html> 即可获得可视化结果。

调用方式

方法：POST

端口：/test_loading

请求参数

参数名	位置	必填	示例值
Content-Type	header	是	application/json

Body参数数据结构

请求报文主要包含已下五个部分：

- items：待装物品的规格信息
- item_inventory：待装货物的数量清单
- containers：备选容器的规格信息
- container_inventory：备选容器的数量清单
- configuration：调用配置信息

1. 详细数据结构介绍请参见在线文档：[Test Loading - Mangrove Planner Integration \(apifox.cn\)](#).
2. 具体参数含义请参见本文档最后部分

Body参数示例

```
{
  "items": [
    {
      "type": "unit_item",
      "value": {
        "extended_properties": {
          "weight": 1
        }
      }
    }
  ]
}
```

```

    },
    "constraints": {
        "allowed_orientations": [
            0,
            1,
            2,
            3,
            4,
            5
        ]
    },
    "uid": "001",
    "name": "O",
    "exterior_shape": [
        350,
        470,
        300
    ]
}
],
"containers": [
    {
        "type": "shipping_container",
        "value": {
            "uid": "40HC",
            "name": "40HC",
            "exterior_shape": [
                12000,
                2400,
                2700
            ]
        }
    }
],
"inventory_items": [
    {
        "uid": "001",
        "quantity": 50
    }
],
"inventory_containers": [
    {
        "uid": "40HC",
        "quantity": 999
    }
],
"configuration": {
    "global_constraints": {
        "min_support_ratio": {
            "enabled": true,
            "default_min_support_ratio": 0.8
        },
        "container_weight_bearing": {
            "enabled": true,

```

```

        "max_weight_bearing": 30000
    },
    "small_on_big": {
        "enabled": false,
        "tolerance": 20
    },
    "light_on_heavy": {
        "enabled": false,
        "tolerance": 10.3
    }
}
}
}

```

返回体报文数据结构

本API原则上将在返回报文中包含所有的原始输入信息，并在其基础上添加装载结果信息。一个典型的成功返回报文将包含以下内容：

- `message`：返回报文主要信息
 - `items`：输入的待装物品的规格信息，此部分内容将与输入报文一致
 - `item_inventory`：待装货物的数量清单，此部分内容将与输入报文一致
 - `containers`：备选容器的规格信息，此部分内容将与输入报文一致
 - `container_inventory`：备选容器的数量清单，此部分内容将与输入报文一致
 - `configuration`：调用配置信息，此部分内容将与输入报文一致
 - `loading_result`：新添加的装载结果信息
 - `remaining_items`：新添加的剩余货物信息，是本次装载计算后未被成功装入的剩余货物
 - `remaining_containers`：新添加的剩余容器信息，是本次装载计算后未使用的备选容器
- `protocol_type`：API协议信息，调试用
- `version`：API版本信息，调试用

1. 详细数据结构介绍请参见在线文档：[## 返回成功示例](https://apifox.cn/Test>Loading - Mangrove Planner Integration (apifox.cn).
2. 具体参数含义请参见本文档最后部分

</div>
<div data-bbox=)

状态码：200

```

{
  "message": {
    "configuration": {
      "global_constraints": {
        "container_weight_bearing": {
          "enabled": true,
          "max_weight_bearing": 30000
        },
        "light_on_heavy": {
          "enabled": false,
          "tolerance": 10.3
        },
        "min_support_ratio": {
          "default_min_support_ratio": 0.8,
          "enabled": true
        }
      }
    }
  }
}

```

```

    },
    "small_on_big": {
      "enabled": false,
      "tolerance": 20
    }
  }
},
"containers": [
  {
    "type": "shipping_container",
    "value": {
      "exterior_shape": [
        12000,
        2400,
        2700
      ],
      "name": "40HC",
      "uid": "40HC"
    }
  }
],
"items": [
  {
    "type": "unit_item",
    "value": {
      "constraints": {
        "allowed_orientations": [
          0,
          1,
          2,
          3,
          4,
          5
        ]
      },
      "extended_properties": {
        "weight": 1
      },
      "exterior_shape": [
        350,
        470,
        300
      ],
      "name": "0",
      "uid": "001"
    }
  }
],
"loading_result": [
  {
    "container_uid": "40HC",
    "loaded_items": [
      {
        "item_uid": "001",

```

```

        "location": [
            0,
            0,
            0
        ],
        "orientation": 3,
        "row_column_layer": [
            3,
            4,
            4
        ]
    },
    {
        "item_uid": "001",
        "location": [
            0,
            1200,
            0
        ],
        "orientation": 3,
        "row_column_layer": [
            1,
            2,
            1
        ]
    }
]
}
],
"protocol_type": "heuristic",
"remaining_containers": [
    {
        "quantity": 998,
        "uid": "40HC"
    }
],
"remaining_items": []
},
"protocol_type": "mangrove_planner_integration",
"version": "mangrove_planner_integration-0.2.0"
}

```

调用失败示例

```

{
    "http_code": 400,
    "http_response": "{\"error\":\"[json.exception.out_of_range.403] key 'items' not found\", \"protocol_type\":\"mangrove_planner_integration\", \"version\":\"mangrove_planner_integration-0.2.0\"}"
}

```

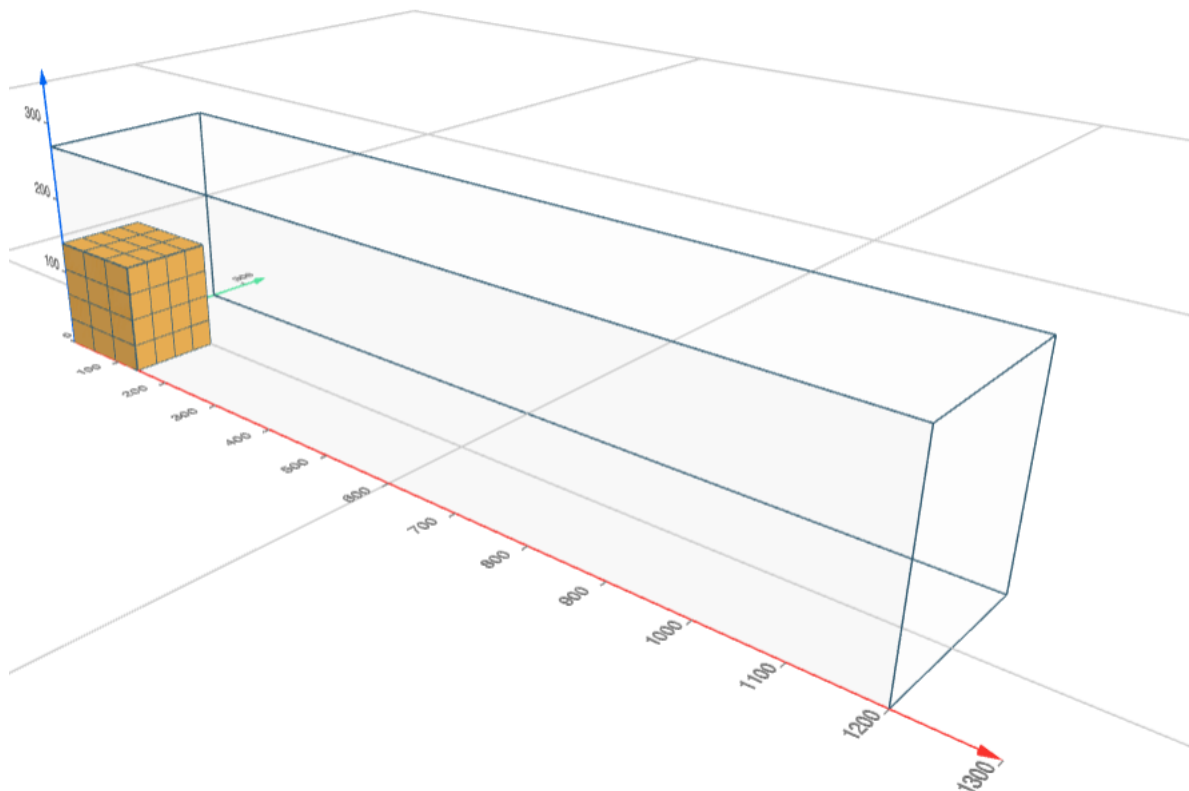
参数含义介绍

参数引用关系

1. 此版本API对输入货物的规格信息和数量实行分开管理，其中：
 1. "items" 字段包含所有待装货物的属性信息，每种货物以 "value::uid" 进行区分，请保持每种货物的uid唯一；
 2. "inventory_items" 字段表示待装货物的数量清单，每种货物以 {"uid": "quantity"} 的方式指定具体的数量，其中 "uid" 表示具体是哪种货物，quantity 表示该种货物的数量，算法将根据货物uid在 "items" 字段中查询货物规格详情，请保证在这里输入正确的货物uid
2. 此版本API对输入容器的规格信息和数量实行分开管理，其中：
 1. "containers" 字段包含所有备选容器的属性信息，每种容器以 "value::uid" 进行区分，请保持每种容器的uid唯一；
 2. "inventory_containers" 字段表示备选容器的数量清单，每种容器以 {"uid": "quantity"} 的方式指定具体的数量，其中 "uid" 表示具体是哪种容器，quantity 表示该种容器可以使用的数量，算法将根据容器uid在 "containers" 字段中查询容器规格详情，请保证在这里输入正确的容器uid
 3. **请注意**，算法将不会使用超过用户指定的容器数量进行装载，多余的货物将被作为剩余货物返回
3. 此版本API的返回报文保留所有输入报文信息，并在其基础上添加以下字段：
 1. "loading_result": 一个列表，表示成功装载的货柜结果，每个列表单元是一个整柜，其中进一步给出了每种货物的摆放朝向和位置
 2. "remaining_items": 一个列表，表示未成功装载的剩余货物，每个列表单元是一种货物及其剩余数量；对于每种输入货物，"inventory_items" 中的数量应该等于返回体中 "loading_result" 与 "remaining_items" 中对应货物的数量之和
 3. "remaining_containers": 一个列表，表示未被使用的剩余容器，每个列表单元是一种货柜及其剩余数量；，对于每种输入容器，"inventory_containers" 中的数量等于返回体中 "loading_result" 与 "remaining_containers" 中对应容器的数量之和

其他参数详情，请参考在线文档：[## 坐标系](https://apifox.cn/Test>Loading - Mangrove Planner Integration (apifox.cn)</p></div><div data-bbox=)

注意：请在<https://aparecium.doraclp.cn/index.html>中进行返回结果的可视化



本协议中所有空间三维坐标均以[X, Y, Z]直角坐标系表示，其中：

- X方向为集装箱长度方向，如图所示从集装箱内部向箱门一侧延申
- Y方向为集装箱宽度方向，如图所示从集装箱内部内部（背对箱门）从左向右延申
- Z方向为集装箱高度反向，如图所示从集装箱内部地面向柜顶延申

以40尺集装箱为例，假设箱长12米，宽2.4米，高2.7米，则：

- 集装箱内部**左后下**角落(背对箱门)为坐标系原点[0, 0, 0]
- 集装箱外侧箱门处右上角坐标为[12000, 2400, 2700]

货物尺寸与旋转方向

每个输入货物可以在 "items" 列表元素的 "value"::"exterior_shape" 字段中声明货物的尺寸。其中，列表中的三个数字分别代表沿x, y, z三个方向的货物长、宽、高。如下所示，货物的长为350毫米，宽为470毫米，高为300毫米。

请注意，货物后续朝向的定义严格依附于货物原始的长宽高，尤其是货物的高度，请按实际货物高度录入

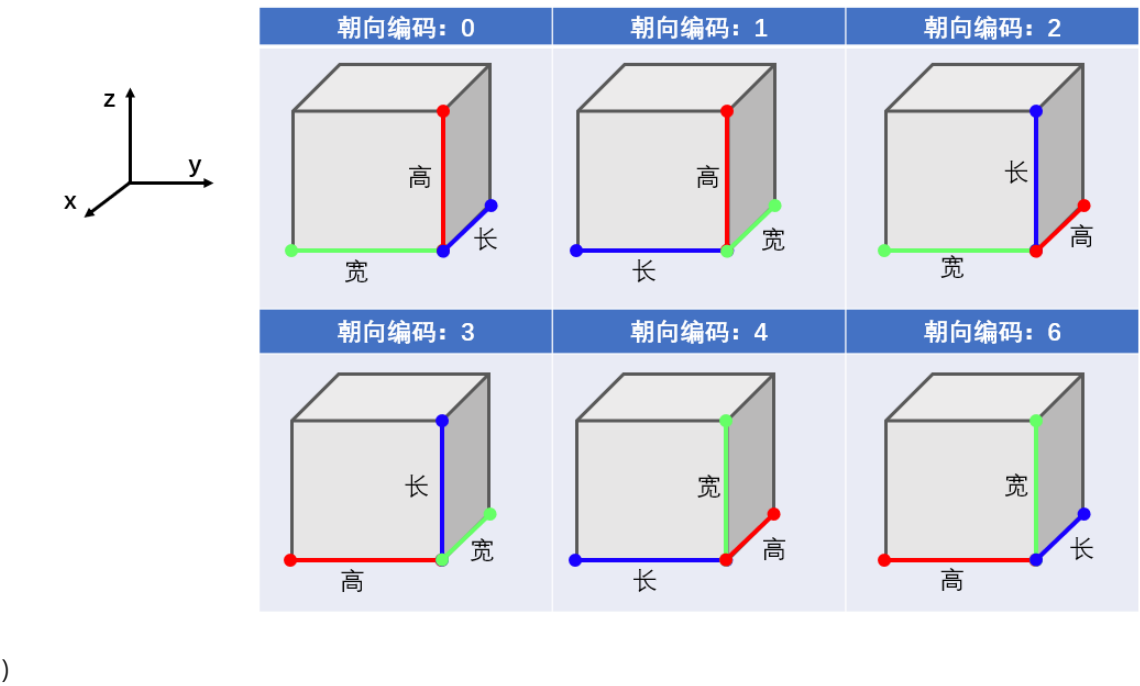
```
"items": [  
  {  
    "type": "unit_item",  
    "value": {  
      "extended_properties": "...",  
      "constraints": {  
        "allowed_orientations": [  
          0,  
          1,  
          2,  
          3,  
          4,  
        ]  
      }  
    }  
  ]  
]
```

```
5
    ]
  },
  "uid": "...",
  "name": "...",
  "exterior_shape": [
    350,
    470,
    300
  ]
}
]
```

在定义货物尺寸后，每个立方形的货物理论上存在6种旋转方向，分别以朝向编码0~5表示，如下图所示。其中，朝向编码0表示输入长宽高的原始朝向，其余5种朝向是原始朝向旋转后的结果。例如：

- 朝向编码1表示"保持货物原始高度边不变，水平旋转货物一次"
- 朝向编码2表示"保持货物原始宽度边不变，前后倾倒货物一次"

以此类推。



每个输入货物可以在 "items" 列表元素的 "value"::"constraints"::"allowed_orientations" 字段中声明该类货物允许的朝向，算法将从此列表选取最终的货物朝向姿态。例如：

- 如果货物可以以任意朝向姿态装载，则 "allowed_orientations": [0,1,2,3,4,5]
- 如果货物仅能水平旋转，不可倾倒（如托盘货），则 "allowed_orientations": [0,1]

以此类推

指定容器

请注意：本版本API不直接支持多柜型的混合计算。

每次调用时，用户需在 inventory_containers 字段中声明使用**1种**容器规格进行装载。算法将根据您指定的容器规格，返回零个或多个柜的装载结果。如果 inventory_containers 字段中声明了多种容器，只有第一种容器会被实际使用。

容器尺寸

用户可以在 "containers" 字段中录入备选容器的规格信息，每种备选容器可以在 "containers" 列表元素的 "value"::"exterior_shape" 字段中声明容器的尺寸。

其中，列表中的三个数字分别代表沿x, y, z三个方向的货物长、宽、高。如下所示，容器的长为12米，宽为2.4米，高为2.7米。

```
"containers": [  
  {  
    "type": "shipping_container",  
    "value": {  
      "uid": "40HC",  
      "name": "40HC",  
      "exterior_shape": [  
        12000,  
        2400,  
        2700  
      ]  
    }  
  }  
]
```

容器数量

用户可在 "inventory_containers" 中指定备选容器的可用数量，例如：

```
"inventory_containers": [  
  {  
    "uid": "40HC",  
    "quantity": 10  
  }  
]
```

以上设定表示**最多使用**10个uid为40HC的容器进行装载。实际装载结果取决于货物的量，示例如下：

1. 如果输入货物只需4个货柜就能装完（货少柜多），则在返回体的 "loading_result" 中给出4个40HC的装载结果，并在返回体的 "remaining_containers" 中返回剩余的6个40HC
2. 如果输入货物需12个40HC才能装下（货多柜少），则在返回体的 "loading_result" 中给出10个40HC的装载结果，并在返回体的 "remaining_containers" 中返回剩余的0个40HC，没有装入的货物，将在返回体的 "remaining_items" 中给出

算法配置

用户可在 "configuration" 字段中配置算法的控制参数，其中包括：

- 所有输入备选容器的最大承重能力: "container_weight_bearing"
- 货物最小底面积支撑比例: "min_support_ratio"
- 是否允许货物大不压小: "small_on_big"
- 是否允许货物重不压轻等: "light_on_heavy"

其中，每个控制参数至少包含一个 "enabled" 的布尔成员，控制该控制参数是否生效，此外还可能包含其他参数成员用来传递具体的控制参数。

例如控制“最小底面积支撑比例”的示例参数如下，只有当 "enabled" 为 true 时此约束条件才会生效，并且仅允许底面积支撑比例80%以上的货物摆放结果

```
"min_support_ratio": {
  "enabled": true,
  "default_min_support_ratio": 0.8
},
```

详细解释说明，请参考在线文档：[Test Loading - Mangrove Planner Integration \(apifox.cn\)](https://apifox.cn/Test-Loading-Mangrove-Planner-Integration)

如需其他类型的约束条件或功能支持，请联系support.aiot@dorabot.com获得进一步解释说明

装载结果解读

1. 待装货物 `inventory_items` 在返回结果中有两种可能：

1. **被成功装载的货物**：其信息将会出现在 `loading_result` 字段中
2. **未被成功装载的货物**：其信息将会出现在 `remaining_items` 字段中，其中 `remaining_items` 的数据结构与 `inventory_items` 类似，通过 `{uid:quantity}` 的对象记录货物的身份和数量信息

2. 备选容器 `inventory_containers` 在返回结果中有两种可能：

1. **被成功使用的备选容器**：其信息将会出现在 `loading_result` 字段中
2. **未被成功使用的备选容器**：其信息将会出现在 `remaining_containers` 字段中，其中 `remaining_containers` 的数据结构与 `inventory_containers` 类似，通过 `{uid:quantity}` 的对象记录容器的身份和数量信息

3. `loading_result` 包含具体的装载结果，其组织方式如下：

1. `loading_result` 是一组对象的列表，其中每个对象是使用相同容器规格的装载结果，例如 `{"container_uid": "40HC", loaded_items: [...]}`，其中
 1. `"container_uid": "40HC"` 表示这组装载结果都是使用uid为40HC的容器，具体容器规格可在 `containers` 字段中查询
 2. `loaded_items` 是这组装载结果具体的货物具体的摆放信息
2. 具体的，由于同一个柜型可能返回多个整柜结果，因此 `loaded_items` 是一个二重列表，其中
 1. 第一层列表的每个元素是一个整柜，`loaded_items` 包含几个元素，则表示这个柜型返回了几个整柜结果
 2. 第二层列表的每个元素是一组货物的具体摆放方式，二层列表有几个元素，可大致理解为该整柜中包含了几次装载动作
 3. 在第二层列表的每次装载动作中，每一批规格相同（`uid` 相同），朝向一致（`orientation` 相同的）的货物会被组合成一个更大的**块**进行集中放置，每个块可能在前后方向上有x排，左右方向上有y列，上下方向上有z层（其中x,y,z都是整数），这个组合成块的信息被记录在 `row_column_layer` 的字段中（`{"row_column_layer": [x,y,z]}`），这个块的左后下角将放置在由字段 `location` 描述的坐标位置

下面给出一个完整的示例：

```
"loading_result": [
{
  "container_uid": "40HC",
  // 本对象包含所有柜型uid为40HC的所有
  // 装载结果
}
```

```

    "loaded_items":[
        // loaded_items是个二重列表，第一层
        // 列表是返回的整柜结果
        [{...},{...},...],
        // 返回的第一个柜子
        [
            {
                // 一个货物块的具体放置信息
                // 货物的uid编号为"001",可据此
                "item_uid":"001",
                // 货物被选用的朝向编号，具体请参
                "orientation":3,
                // 见上文货物朝向示意
                "row_column_layer":[3,4,5],
                // 这个货物块的前后有3排，左右有4
                "location":[0,0,0],
                // 这个货物块的左后下角坐标是
                [0,0,0]
            },
            // 下一个货物块摆放信息
            {...},
            ...
        ],
        // 返回的第二个柜子
        // 返回的第三个柜子
        [{...},{...},...],
        ...
    ]
}

```

服务声明

本API服务由深圳蓝胖子机器智能有限公司提供，如您在使用过程中有任何问题，请联系 support.aiot@dorabot.com