# 数据

知识点：构造类

class 类名｛

constructor(形参列表){

//构造函数

//属性列表

}

//方法列表

方法名(){

}

｝

构造对象

var 对象名=new 类名（实际参数列表）

## 数据结构

商品类-- Product

|  |  |
| --- | --- |
| id---编号  title—名称  price—价格  imgSrc---图片地址 | class product {  constructor(id, title, imgSrc, price) {  this.id = id;  this.title = title;  this.imgSrc = imgSrc;  this.price = price;  }  } |

订单类—Order

|  |  |
| --- | --- |
| id---商品编号  title—商品名称  price—商品价格  imgSrc--商品图片地址  qty--数量  selectStatus—选择状态 | class Order {  constructor(product, qty, selectStatus) {  this.id = product.id;  this.title = product.title;  this.imgSrc = product.imgSrc;  this.price = product.price;  this.qty = qty;  this.selectStatus = selectStatus;  }  }; |

购物车类—CartData—构建时是空购物车状态

|  |  |
| --- | --- |
| 订单列表  总样本数  总件数  总价格 | class CartData {  constructor() {  this.orderList = new Array();  this.totalQty = 0;  this.totalAmount = 0;  this.units = 0;  }  } |

# 购物车数据操作

知识点：

// localStorage.setItem("key","value");//存储变量名为key，值为value的变量

// localStorage.key = "value"//存储变量名为key，值为value的变量

// localStorage.getItem("key");//获取存储的变量key的值

// localStorage.key;//获取存储的变量key的值

// localStorage.removeItem("key")//删除变量名为key的存储变量

购物车类（操作）

|  |  |
| --- | --- |
| setDataToLocalSatorge:数据写入localStorage  getDataFromLocalSatorge :读取数据  addToCart(order):将订单加入购物车 | Class ShoppingCart{  setDataToLocalSatorge(cartData) {  }  getDataFromLocalSatorge()｛  ｝  addToCart(order)｛  ｝  } |

## 数据写入localStorage—ShoppingCart. setDataToLocalStorage()

### 技术

（1）将数据（字符串）写入本地存储 localStorage.setItem("key","value");//存储变量名为key，值为value的变量

（2）变量转换为JSON字符串JSON.stringify(变量名)

### 解题思路

入口参数 cartData(购物车数据)

1. 将cartData转换为JSON字符串

JSON.stringify(cartData)

1. 将cartData字符串写入localStorage

localStorage.setItem("key","value");//存储变量名为key，值为value的变量

|  |
| --- |
| // 将购物车数据写入本地存储中  setDataToLocalSatorge(cartData) {  //清除原有存储写入新列表  localStorage.removeItem('lzzyCart');  //写入本地存储  localStorage.setItem('lzzyCart', JSON.stringify(cartData));  } |

## 从localStorage读取数据

### 技术

（1）从本地存储中获取指定key的值-- localStorage.getItem("key")

（2）JSON字符串转JSON变量JSON.parse(字符串)

### 解题思路

1. localStorage.getItem("key");//获取存储的变量key的值
2. 判断读取到到数据是否为null或者‘’
   1. 不为空转换json字符串🡪js对象 JSON.parse(字符串)
   2. 为空，创建新购物车数据对象
3. 返回js对象

|  |
| --- |
| // 从本地存储中获取购物车数据  getDataFromLocalStorage() {  let lzzyCart = localStorage.getItem('lzzyCart');  // 判断购物车是否为空  if (lzzyCart == null || lzzyCart == '') {  return new CartData();  }  else {  return JSON.parse(lzzyCart);  }  } |

## 订单数据加入购物车（本地存储）

### 技术

for-in 遍历数组或对象

for (key in object) {

object[key]//如果object是对象则object[key]等价于引用object的key成员 等价于object.key

}

for (key in object) {

object[key]//如果object是数组则object[key] 等价于object数组等第key个元素

}

### 解题思路

1. 读取本地存储中的购物车数据 cartData=this. getDataFromLocalSatorge()
2. 把订单加入订单列表中
3. 完善总计等数据
4. 将新购物车数据写入本地存储 this. setDataFromLocalSatorge(cartData)

### 代码实现

|  |
| --- |
| addToCart(order) {  // 从本地存储中获取购物车的数据  let cartData = this.getDataFromLocalStorage();  // 获取购物车json数据中的订单列表  let orderList = cartData.orderList;  //设置标志位判断是否为购物车新商品，默认为是新商品  let isNewProduct = true;  // 遍历订单列表，判断新加入商品是否在购物车中  for (let i in orderList) {  if (order.id == orderList[i].id) {  // 若新加入订单商品已经在购物车中，则变更订单列表中对应商品数量，且变更新商品标志位  orderList[i].qty += order.qty;  isNewProduct = false;  break;  }  }  //如果是新商品  if (isNewProduct) {  // 购物车总样本+1  cartData.units++;  // 导入新商品置入购物车  orderList.push(order);  }  //修改购物车总金额及商品总数量  cartData.totalAmount += order.qty \* order.price;  cartData.totalQty += order.qty;  // 写入localStorage  this.setDataToLocalSatorge(cartData);  } |

## 清空购物车

### 技术

移除指定键位的本地存储localStorage.removeItem(“key”);

清除所有本地存储 localStorage.clear

### 解题思路

移除指定键位的本地存储

### 代码实现

|  |
| --- |
| // 清空购物车（移除本地存储购物车项）  clearCart() {  localStorage.removeItem('lzzyCart');  } |

## 获取选中订单的总件数

### 解题思路

1. 获取购物车数据
2. 获取订单列表
3. 遍历订单列表，逐一判断每个订单的选择状态
   1. 选中的，累加订单数量
   2. 未选中，就不统计该订单数量
4. 返回选中状态的总数量

### 代码实现

|  |
| --- |
| // 获取选中商品的总数量  getSelectedQty() {  let cartData = this.getDataFromLocalStorage();  let orderList = cartData.orderList;  let selectedQty = 0;  for (const key in orderList) {  if (orderList[key].selectStatus) {  selectedQty += orderList[key].qty;  }  }  return selectedQty;  } |

## 获取选中订单的总价格

### 解题思路

1. 获取购物车数据
2. 获取订单列表
3. 遍历订单列表，逐一判断每个订单的选择状态
   1. 选中的，累加 （订单数量\*订单价格）
   2. 未选中，就不统计
4. 返回选中状态的总价格

### 代码实现

|  |
| --- |
| // 获取选中商品的总价格  getSelectedAmount() {  let cartData = this.getDataFromLocalStorage();  let orderList = cartData.orderList;  let selectedAmount = 0;  for (const key in orderList) {  if (orderList[key].selectStatus) {  selectedAmount += orderList[key].qty \* orderList[key].price;  }  }  return selectedAmount;  } |

## 设置某个订单（id）的选择状态（selectStatus）

解题思路：

1. 获取购物车数据
2. 获取订单列表
3. 遍历订单列表，逐一判断每个订单的id是否等于指定id（形式参数）
   1. 找到对应id,设置选择状态selectStatus（形式参数） 结束循环
   2. 没找到，继续遍历直到订单列表遍历完成

## 删除某个指定（id）订单

解题思路：

1. 获取购物车数据
2. 获取订单列表
3. 遍历订单列表，逐一判断每个订单的id是否等于指定id（形式参数）
   1. 找到对应id,删除指定id的订单， 修改总价格，总样本数，总数量

* 1. 没找到，继续遍历直到订单列表遍历完成，提示没有找到指定id

## 某个指定订单数量减1

## 某个指定订单数量加1