# JS Advanced: Exam 18 March 2018

Problems for exam preparation for the [“JavaScript Advanced” course @ SoftUni](https://softuni.bg/courses/javascript-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/974/>.

# Problem 3. Payment Processor (Simple Class)

Write a JavaScript class PaymentProcessor that keeps information about a **collection** of payments with their **ID**, **name**, **type** and **value**.

|  |
| --- |
| **class** PaymentProcessor {  *//* ***TODO: implement this class*** } |

The class constructor receives an **optional object parameter** that defines its behavior (see below for details). Implement the following features:

Function registerPayment(id, name, type, value) – validate input parameters and save the payment; throw and error if the validation fails (see below for details)

Function deletePayment(id) – removes the requested payment; throw an error if it’s not found

Function get(id) – returns a string, containing information about the requested payment (see examples for formatting details); throw an error if it’s not found

Function setOptions(options) – modify the processor’s options

Function toString() – return a string, containing a summary about all recorded payments (see examples for formatting details)

The options parameter that the **constructor** takes is an **object** with the following format:

{

types: [String],

precision: Number

}

When processing **new options** (either through the **constructor** or the setOptions function), only replace the properties that are supplied – e.g. if only types are given, replace the existing types value (entire array), leaving precision unchanged. If not specified, use the following **default values**:

{

types: ["service", "product", "other"],

precision: 2

}

A **valid** payment will have an **ID** and **name** that are **non-empty strings**, a **value** that is a **number** and a **type** that is listed in options. If a payment with the **same ID** already **exists**, consider the new one **invalid**. For any discrepancy, **throw an error** and **ignore** the payment. When recording the payment, round its value to the number of decimal places, specified in options.precision.

***Scroll down for examples and constraints.***

### Examples

|  |
| --- |
| Sample code usage |
| *// Initialize processor with default options* **const** generalPayments = **new** PaymentProcessor(); generalPayments.registerPayment(**'0001'**, **'Microchips'**, **'product'**, 15000); generalPayments.registerPayment(**'01A3'**, **'Biopolymer'**, **'product'**, 23000); ***console***.log(generalPayments.toString());  *// Should throw an error (invalid type)* generalPayments.registerPayment(**'E028'**, **'Rare-earth elements'**, **'materials'**, 8000);  generalPayments.setOptions({**types**: [**'product'**, **'material'**]}); generalPayments.registerPayment(**'E028'**, **'Rare-earth elements'**, **'material'**, 8000); ***console***.log(generalPayments.get(**'E028'**)); generalPayments.registerPayment(**'CF15'**, **'Enzymes'**, **'material'**, 55000);  *// Should throw an error (ID not found)* generalPayments.deletePayment(**'E027'**); *// Should throw an error (ID not found)* generalPayments.get(**'E027'**);  generalPayments.deletePayment(**'E028'**); ***console***.log(generalPayments.toString());  *// Initialize processor with custom types* **const** servicePyaments = **new** PaymentProcessor({**types**: [**'service'**]}); servicePyaments.registerPayment(**'01'**, **'HR Consultation'**, **'service'**, 3000); servicePyaments.registerPayment(**'02'**, **'Discount'**, **'service'**, -1500); ***console***.log(servicePyaments.toString());  *// Initialize processor with custom precision* **const** transactionLog = **new** PaymentProcessor({**precision**: 5}); transactionLog.registerPayment(**'b5af2d02-327e-4cbf'**, **'Interest'**, **'other'**, 0.00153); ***console***.log(transactionLog.toString()); |
| Corresponding output |
| Summary:  - Payments: 2  - Balance: 38000.00  Details about payment ID: E028  - Name: Rare-earth elements  - Type: material  - Value: 8000.00  Summary:  - Payments: 3  - Balance: 93000.00  Summary:  - Payments: 2  - Balance: 1500.00  Summary:  - Payments: 1  - Balance: 0.00153 |

### Constraints

* Your class will be tested with both **valid and invalid parameters** and should validate the input to registerPayment, deletePayment, get and setOptions
* Your class will be tested with **only valid options**

### Submission

Submit **only** your class PaymentProcessor.

### Hint

To create a string, that contains a line break, use the special character '\n'.