レポート問題：2.16版

- 宣伝：金融機関A社では、ただ今年利23%で貸し出しをおこなっております。電話でお申し込みいただければ、簡単な審査で即日300万円まで、ご指定の銀行へお振り込みいたします。
- 参考：年23%は、一見他社に比べ圧倒的に低いように見えるが、そのからくりを見破れる？
- 設問：100万円を借りると、10年後には利子元金合わせていくら返済する必要があるでしょうか。その試算のための返済シミュレーション・ソフトを、依頼されて作ってみましょう。
- 上の設問以外の開発方針を自分で立てて、それに基づいてレポートすれば、さらに高い評価をします。

補足：各社経営戦略

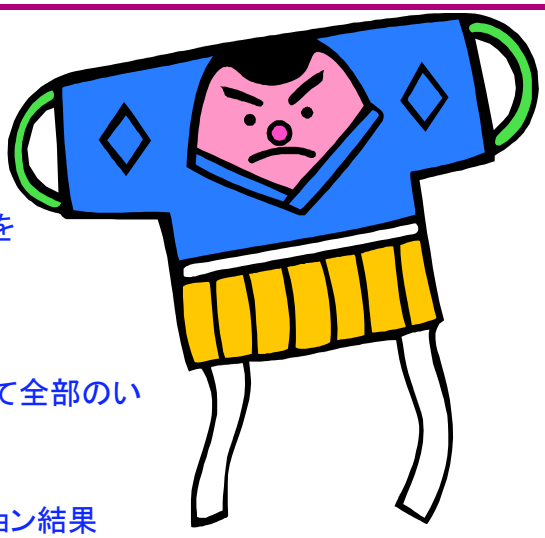
- 業績好調A社の場合：支店を、毎年20%の割合で増やす。それに対して人員増でなく、設備投資で対処したい。
- 業績不調B社の場合：支店は増やさず、人員を3割カットすることで、赤字解消を図りたい。設備投資も前年比7割カットで行きたい。
- 堅実路線C社の場合：支店数は業界伸び率平均にとどめ、利益率を現状の2%から、3年間かけて4%まで引き上げたい。そのための施策は積極的におこなう。

レポート作成要領（例）

- 例として添付されたものを元に改良して見る。
- または、自分で設問を考え、独自開発方針から始めるのでも良い。
- 一部でも、未完成でもかまわない。 自分の頭で考えているかどうかを重視する。

レポート提出要領

- 提出
  - 期限：**2016/10/8土曜日必着**
  - 提出先：佐藤和夫 [kazuo.satoh.1＊vc.ibaraki.ac.jp]
    ＊は@（アット）に変えて宛先としてください。
  - 「受け取りました」返事が翌日返ってこないときは、再送を
- 提出物の例
  自分の頭で考えていれば、
  一部の文書でも、未完成でもかまいません。
  設計、設計およびプログラミング、設計から試験まで通して全部のいずれかを選択してください。
  - **設計**: 開発方針、外部設計書、内部設計書
  - **プログラミング**: プログラム・コード、コード・インスペクション結果
    （設計問題記録、プログラム問題記録）
  - **試験**: 単体試験、結合試験、総合試験局面のテストケースおよび試験結果
    （設計問題記録、プログラム問題記録）



お願い：文書はpdf,xls,rtf,txt,c,c++,java,javascriptいずれかでご送付ください。

レビューをしたら
この用紙を活用！

設計問題記録

問題記述	発生日	影響度	解決予定日	解決責任者	解決日	誤り原因分類A,B	原因記述

定義:
　誤り:文書間の論理の食い違い・ 問題:誤りの候補・ 欠陥:外部設計と内部設計以降の文書との食い違い・
影響度:
　大:この問題を解決しないと、他部品の設計／結合試験／総合試験が出来ない。
　中:他部品設計／結合試験／総合試験への影響はないが、当該部品の設計／結合試験／総合試験が出来ない。
　小:影響軽微
解決日:
　解決済みの問題は、別リストとして保管する。
原因分類:
　A－1. 開発方針誤り 2. 外部設計誤り 3. 内部設計誤り
　B－
　　1.開発方針見落とし
　　2.外部設計見落とし
　　3.外部設計理解不足
　　4.外部設計変更連絡漏れ
　　5.外部設計検討不足
　　6.曖昧表現
　　7.その他

コード・インスペクション、
試験をしたら
この用紙を活用！

プログラム問題記録

問題記述	発生日	影響度	解決予定日	解決責任者	解決日	誤り原因分類A,B	原因記述

定義:
　誤り:文書間の論理の食い違い・ 問題:誤りの候補・ 欠陥:外部設計と内部設計以降の文書との食い違い・
影響度:
　大:この問題を解決しないと、他部品のプログラム／単体試験が出来ない。
　中:他部品プログラム／単体試験への影響はないが、当該部品のプログラム／単体試験が出来ない。
　小:影響軽微
解決日:
　解決済みの問題は、別リストとして保管する。
原因分類:
　A－1. 開発方針誤り 2. 外部設計誤り 3. 内部設計誤り 4. プログラム誤り
　B－
　　1.開発方針見落とし
　　2.外部設計見落とし
　　3.外部設計理解不足
　　4.外部設計変更連絡漏れ
　　5.外部設計検討不足
　　6.曖昧表現
　　7.その他
　　12.内部設計見落とし
　　13.内部設計理解不足
　　14.内部設計変更連絡漏れ
　　15.内部設計検討不足

返済シミュレーション・ソフト

外部設計例

開発方針例

変更点	新業務実施方法	従来の業務実施方法	新業務の長所	新業務の課題
25分以内の振り込み可能とする、 業務プロセスの実現	・顧客DB中心への全社業務プロセス変更	各部門連絡は、FAXもしくは電話。部門内は口頭もしくは紙ベース連絡	最短の振り込み時間実現による競争力強化、 他社は翌日	・業務プロセス改革のため、要全社的調整 ・要新業務プロセス徹底
多部門からの利用可能な、顧客DB中心返済シミュレーション・ソフトの提供	・窓口、経理、与信、債権回収、長期計画部門共通機能提供 ・既存借入との合計回答サービス開始	・窓口、与信は電卓、経理はコンピュータ、その他は山勘 ・窓口は、新規借り入れのみ電卓で計算	・与信精度の向上 ・借りすぎ防止 ・無限さん導入、顧客自身の操作による利便性向上	利子変動への対応のために、顧客DBの再編成要。今回は見送り

データ分析例

WHO	WHEN(CONDITION)	WHAT(ACTION)	WHAT(DATA)

窓口	ランダムに顧客が100人/台/日	窓口PCに入力し、印刷物を手渡し	顧客DB {
			顧客ID、
			本日の金利、
			返済予定日数、
			元金、
			返済額
			}

顧客自身	ランダムに顧客が50人/台/日	「無限さん」に顧客自身で入力し、画面に表示	上に同じ

データ制約

データ制約の種類	ドメイン制約	識別子(主キー)	存在制約	参照制約	多重度制約	導出制約	関連制約	更新制約	処理順序制約
A=顧客ID	{1000000..9100000}、整数	YES。顧客DBを通してユニークであること	必須	参照不可	1	なし	なし	なし	見積もり後、顧客が望む場合は与信システムに処理を引き継ぐ
B=本日の金利	{ <0%..23% }、整数もしくは実数	NO	必須	非該当	1	なし	なし	なし	なし
C=返済予定日数	{0..3650}、整数	NO	必須	非該当	1	なし	なし	なし	なし
E=返済額	{ 0..8桁 }、正の整数	NO	有効なA,B,C存在	非該当	1	E=(1+B/36500)**C 1円未満は切り上げ	なし	なし	なし

返済シミュレーション・ソフト

内部設計例

環境情報からくるBQC・回答例

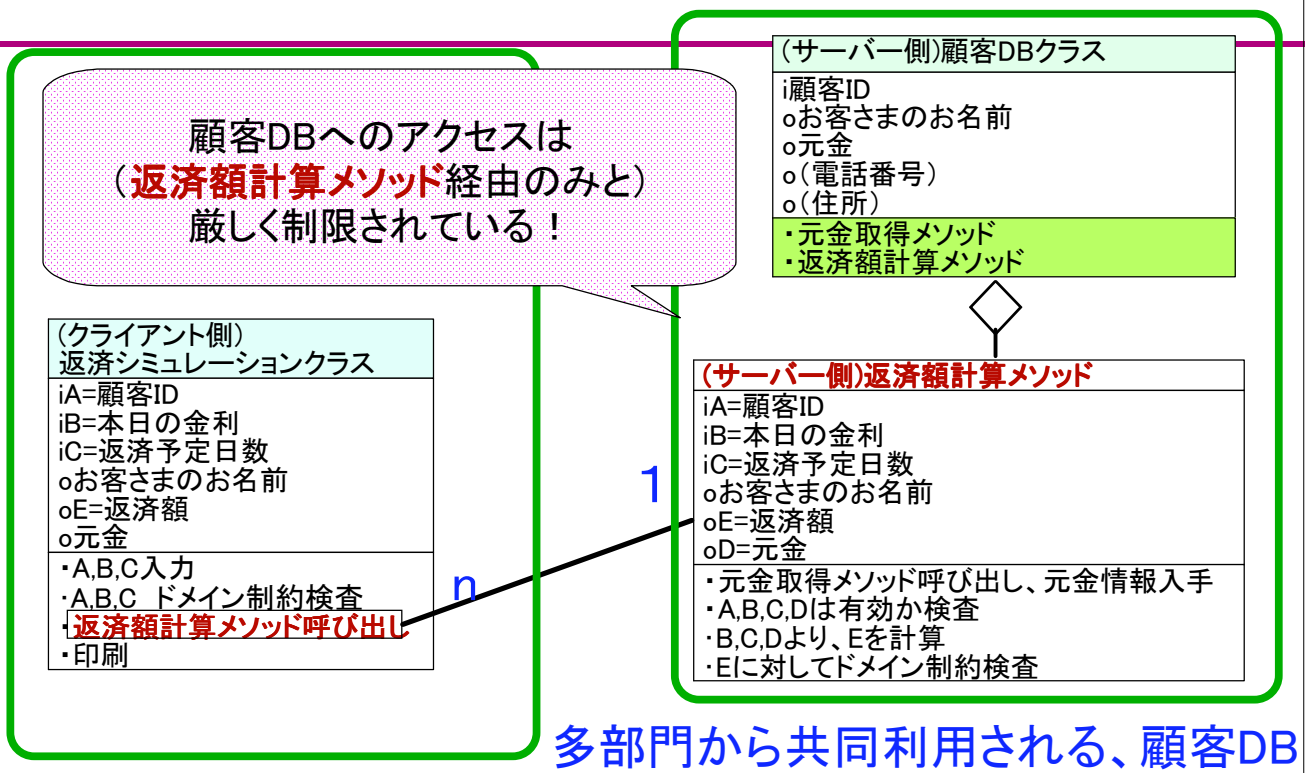
- 稼働時の運用情報、環境情報を明確にし、それから来る例外事項を疑問に浮かべることにより、開発対象ソフトの限界、弱さ(BQC)を捜す。

WHO	WHEN(CONDITION)	WHAT(ACTION)	例外事項への疑問(if then分析)	BQC（内部設計の限界、弱さ)を捜す	対策
窓口	ランダムに顧客が100人/台/日	窓口PCに入力し、印刷物を手渡し	窓口が顧客情報を盗めるか？	盗まれる可能性あり	教育に力を入れる。
			窓口PC故障したらどうなるか？	代替PCおよび無限さんあり	隣のPC、無限さんに誘導
			サーバ故障したらどうなるか？	支店業務停止	電卓で対応
顧客自身	ランダムに顧客が50人/台/日	「無限さん」に顧客自身で入力し、画面に表示	他人の顧客情報利用可能？	偽造・盗難顧客カード利用可能	お手上げ
			無限さん故障したらどうなる？	無限さん代替不可	窓口に誘導
			サーバ故障したらどうなるか？	代替機ないため業務停止	稼働予定を表示
泥棒が、	夜中に、	窓口PCを立ち上げて、顧客情報を盗む 「無限さん」を操作し、顧客情報を盗む	盗めるか？	夜中は運用していない、かつ支店に顧客情報ないため、盗まれる可能性なし	設計変更必要なし

運用情報・環境情報

- 運用時間：朝10時から夜8時（営業時間内）、土曜日曜も稼働
- サーバ：本店に設置。予備機はなし、32ビット演算機
- 窓口PC：各支店に2台以上配置、32ビット演算機
- 無限さん：20支店内に1台以上配置、32ビット演算機

返済シミュレーション・ソフト
内部設計例



内部設計－HIPO

クラスの機能：返済シミュレーションクラス

入力データ	プロセス	出力データ
■ 顧客ID ■ 本日の金利 ■ 返済予定日数	■ **顧客ID,本日の金利,返済予定日数 ドメイン制約検査** ・顧客IDが1000000-910000の範囲の正数 ・本日の金利が0%以下でない、23%を越えない、整数もしくは実数 ・返済予定日数は、0以上の整数 ■ 返済額計算メソッドからお客さまのお名前、返済額、元金をもらう。 ■ それと入力データを併せて印刷する。 ■ 印刷仕様、つぎの項目は全部自由形式とする。 ・ 顧客ID ・ 本日の金利 ・ 返済予定日数 ・ お客さまのお名前 ・ 返済額 ・ 元金	■ 顧客ID ■ 本日の金利 ■ 返済予定日数 ■ お客さまのお名前 ■ 返済額 ■ 元金

プロセス中の記述法：
順序性の規則がある。同じレベルの文章では、上から下へ。
文章中では左から右へ。

内部設計－HIPO

メソッドの機能：返済額計算メソッド

入力データ	プロセス	出力データ
■顧客ID ■本日の金利 ■返済予定日数	■元金取得メソッド呼び出し （入力：顧客ID、出力：お客さまのお名前、元金） ■IF 不明の顧客IDであれば、 エラーE401 //顧客ID有効か検査 ■IF 元金＞0でなければ、 エラーE402 //元金は有効か検査 ■本日の金利、返済予定日数、元金より、返済額を計算 ■返済額に対してドメイン制約検査	■お客さまのお名前 ■返済額 ■元金

プロセスの記述法：
順序性の規則がある。同じレベルの文章では、上から下へ。
文章中では左から右へ。

内部設計－HIPO

メソッドの機能：元金取得メソッド

入力データ	プロセス	出力データ
■顧客ID	■顧客IDを手がかりに顧客DBより、お客さまのお名前、元金を取得 ■ IF 顧客IDが顧客DBに未登録ならば、//顧客IDは有効か検査 エラーコード=1を返す。	■お客さまのお名前 ■元金

プロセスの記述法：
順序性の規則がある。同じレベルの文章では、上から下へ。
文章中では左から右へ。

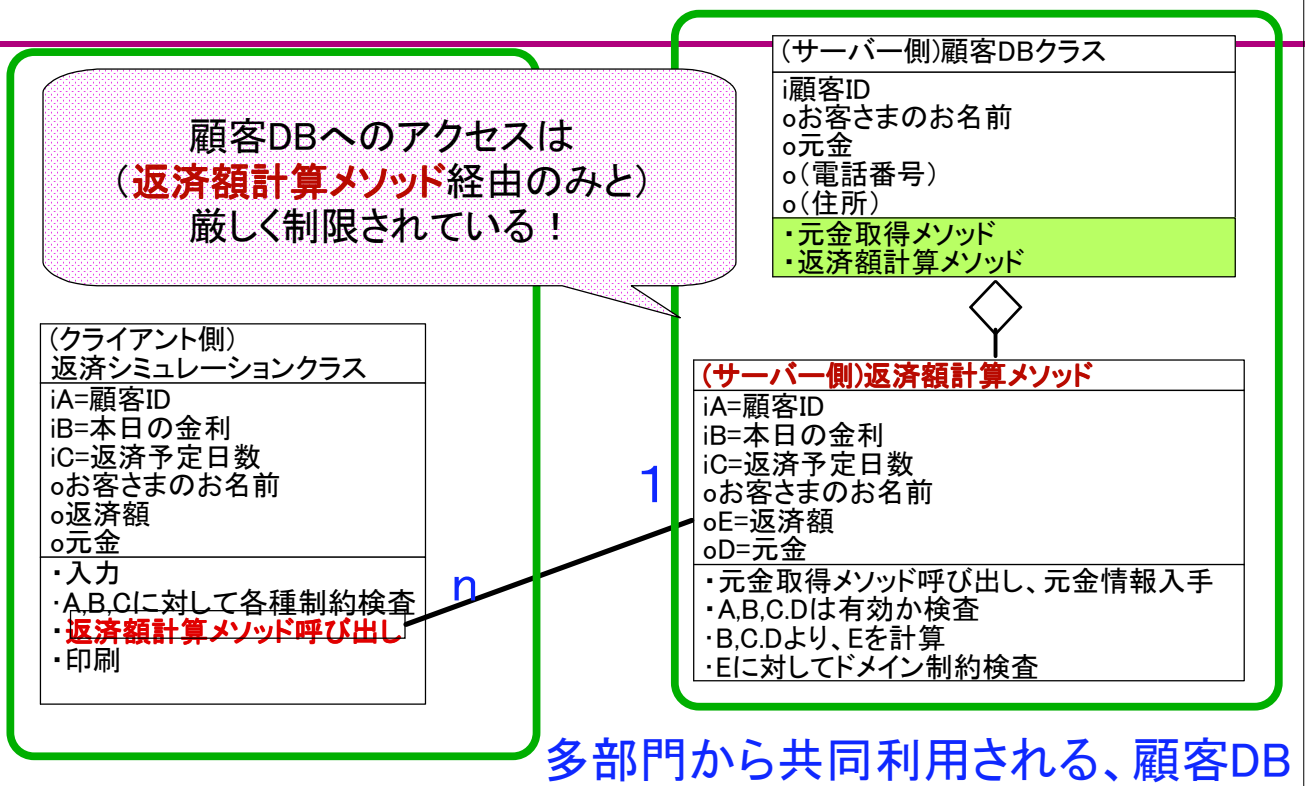
Javaによる、
顧客DB中心のオブジェクト指向プログラミング

Java版返済シミュレーション・ソフト コード例

Readme

- **コード例のありか：**
  - **ダウンロード・サイト** http://www.k4.dion.ne.jp/~skazuo/boo/SD/
  - 「講義で使用するサンプル/プログラム」をクリックしてensyuu.zipをダウンロードする。
  - ensyuu.zipに、Javaで作成されたコード例が格納されている。
  - 適当なディレクトリーに解凍する。
- 学科、総合情報処理センターの環境では：
  - 各PCにあらかじめJDKの導入をしてあるので、コード例はそのまま実行できる。
- 自宅PCの環境では： JDK(Java開発キット)の導入
  - 最新JDK のダウンロードサイト： http://www.oracle.com/technetwork/java/javase/downloads/index.html
  - ダウンロードしたファイルをダブルクリックしてインストールする。
  - JDKのPATH設定をPCへ行う、もしくは返済額計算結果.bat, 顧客DB.bat, 返済シミュレーション.bat, シミュレーション実行.bat, jcpad中の環境変数を、rem行を参考に**jdk1.8.0_91**から変更して設定する。
- コード開発、実行
  - サーバー側-返済額計算結果.batをダブル・クリック(実行)して、返済額計算結果クラスをコンパイルする。
  - サーバー側-顧客DB.batをダブル・クリック(実行)して、顧客DBクラスをコンパイルする。
  - 次に クライアント側-返済シミュレーション.batを実行して、返済シミュレーションクラスをコンパイルする。
  - 最後に クライアント側-シミュレーション実行.batを実行すると、結果がシミュレーション結果.txtに格納される。

Java版コード構成



顧客DBクラス　コード例 (本店側プログラム)

```
import java.io.*;
import java.util.*;
public class 顧客DB {
  public 顧客DB() { };
  //---------------------------------------
  private static class 元金取得結果{
    private String お客様のお名前;
    private long 元金;  };
  private static 元金取得結果 元金取得メソッド
(String 顧客ID){
    元金取得結果 out=new 元金取得結果();
    if (顧客ID.equals("234")) {
      out.お客様のお名前="Satoh Kazuo";
      out.元金=1000000;
    }
    return out;
  }
```

```
public class 返済額計算結果{
  public String お客様のお名前;
  public long 返済額;
  public long 元金;
};
```

```
public static 返済額計算結果 返済額計算メソッド(
String 顧客ID,
double 本日の金利, //(単位:%)
int 返済予定日数   //(単位:日)
) {
  返済額計算結果 out = new 返済額計算結果();
  try{
    元金取得結果 in = 元金取得メソッド(顧客ID);
    long 返済額=in.元金; double 本日の日歩=本日の金利/36500;
    for (int i=1; i<=返済予定日数; i++){
      //日歩利子計算。ただし、1円以下は切り上げ。
      返済額 = (long)(返済額*(1+本日の日歩) +1);
    }
    out.お客様のお名前=in.お客様のお名前;
    out.返済額=返済額;
    out.元金=in.元金;
  }
  catch (Exception e) {
    System.out.println("Exception in 返済額計算メソッド: " + e);
  }
  return out;
 } //end of 返済額計算メソッド
} // end of 顧客DBクラス
```

返済シミュレーションクラス・コード例(支店側プログラム)

```
import java.io.*;
import java.util.*;
class 返済シミュレーション{
  //---------------------------------------------------------------------
  public static void main(String arg[]) {
    try{
      String 顧客ID = arg[0];        //顧客ID
      Double rr=Double.valueOf(arg[1]);
      double 本日の金利=rr.doubleValue();     //本日の金利    （単位：%）
      int 返済予定日数=Integer.parseInt(arg[2]);//期間    （単位：日）
      FileWriter F2= new FileWriter("シミュレーション結果.txt");
      顧客DB db= new 顧客DB();
      返済額計算結果 in = db.返済額計算メソッド(顧客ID,本日の金利,返済予定日数);
      F2.write("顧客ID: "+顧客ID+" お客様のお名前  "+in.お客様のお名前+" 元金 "+in.元金+"円¥r¥n");
      F2.write("返済予定日数"+返済予定日数+"日間; 本日の金利"+本日の金利+"%; 返済額 "+in.返済
額+"円¥r¥n");
      F2.close();
    }
    catch (Exception e) {
      System.out.println("Exception in main: " + e);
    }
  }
  //---------------------------------------------------------------------
} // end of 返済シミュレーションクラス
```

JavaScriptによる、手続き指向プログラミング

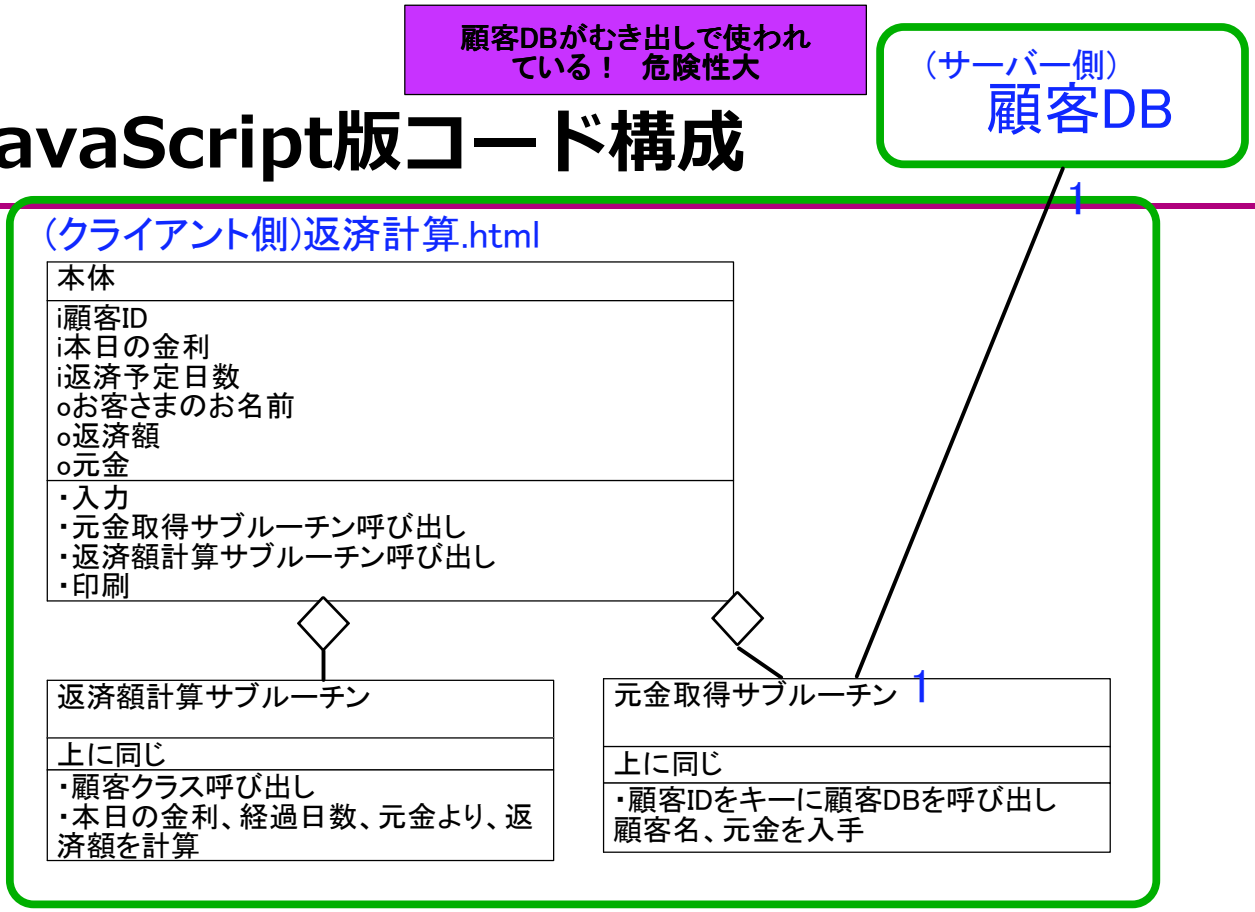
JavaScript版
返済シミュレーション・ソフト
コード例

Readme

- コード例のありか：
  - ダウンロード・サイト　http://www.k4.dion.ne.jp/~skazuo/boo/SD/
  - ensyuu.zipに、JavaScriptで作成されたコード例(返済計算.html)が格納されている。
  - 適当なディレクトリーに解凍する。
- 編集および実行：
  - 実行は、返済計算.htmlをダブル・クリックすればよい。
  - 編集は、適当なテキスト・エディター（NOTEPADなど)やhtmlエディターでやる。

JavaScript版コード構成

顧客DBがむき出しで使われている！　危険性大



返済計算.html
コード例

```
<html>
    <head>    </head>
    <body>
        <script type="text/javascript">
        <!--
 var 顧客ID,r=0,n=0;
 do {顧客ID = prompt("顧客IDを入力してください","?????")}  while (顧客ID=="?????");
 var parm=元金取得サブルーチン(顧客ID);
 var custName=parm[0];
 var m=parm[1];
 document.write ("顧客ID: "+顧客ID+ ", お客様のお名前  "+custName+" 元金￥"+  m+   ">>これでよいですね？<br>");
 do {r = prompt("本日の金利(%)を入力してください",0)} while ( r<=0);
 do {n = prompt("返済予定日数(経過日数)を入力してください",0)} while (n<=0);
 var sum=返済額計算サブルーチン(m,r,n);
 document.write("顧客ID: "+顧客ID +", お客様のお名前  "+custName+" 元金￥"+m+ " 返済予定日数"+n+"日間; 本日の金
利"+r+"%; 返済額￥"+sum );
//--------------------------------
function 元金取得サブルーチン(顧客ID){
 var parm=["",0];
 if (顧客ID=="234") parm=["Satoh Kazuo",1000000];
 return parm;
}
//--------------------------------
function 返済額計算サブルーチン(m,r,n){
 var sum=m, i;
 for (i=1; i<=n; i++) {sum = sum+ Math.floor(sum*(r/36500)+1);}
 return sum;
}
        // -->
        </script>
    </body>
</html>
```

終わり