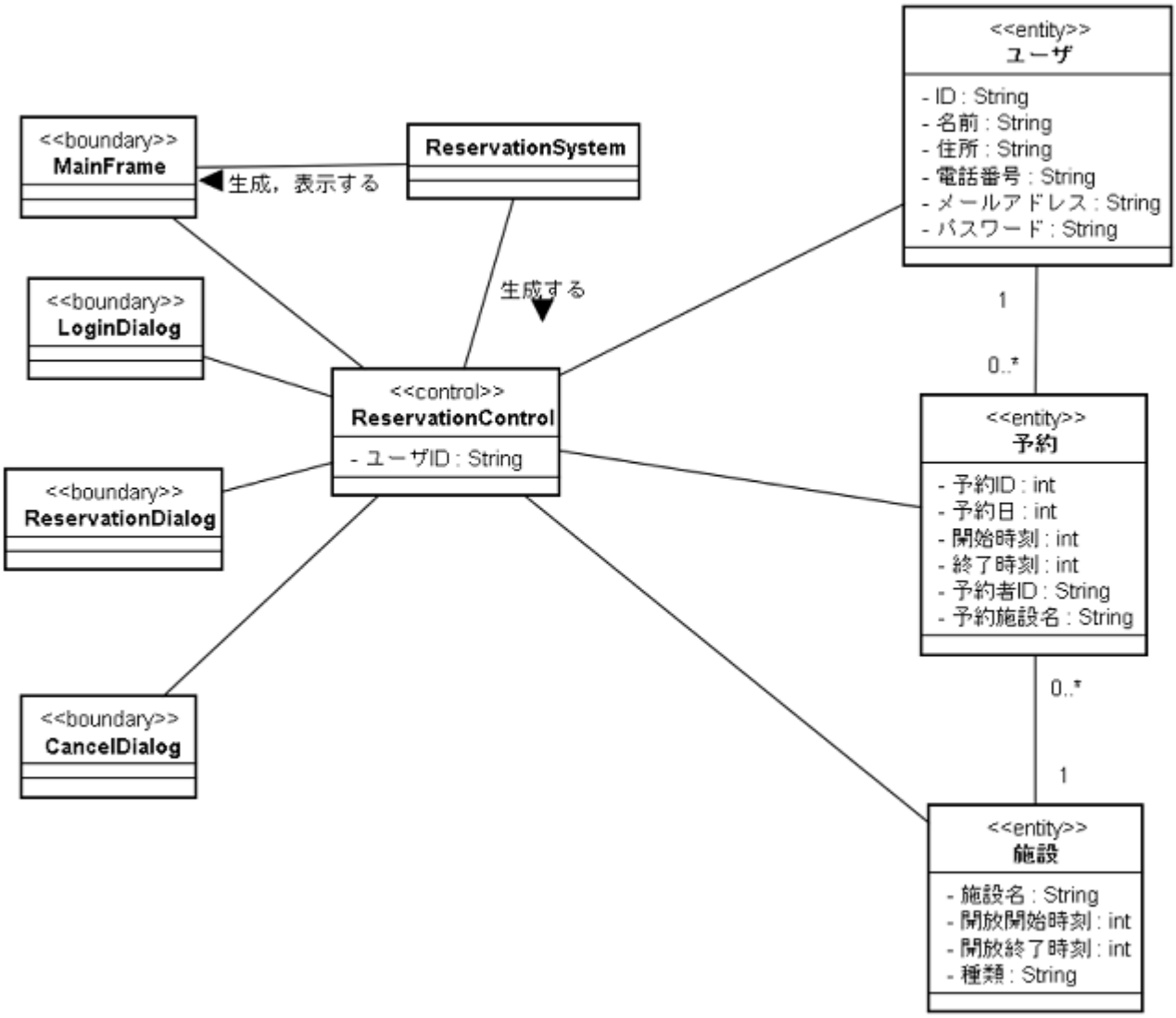
イントロダクション

利用者用サブシステムの最初のバージョンを作成します．今回のバージョンでは，基本的なインターフェースとしてメインの画面を作成し，空き状況の確認機能を実装します．余力がある学生は，施設概要の表示機能も実装しましょう．

作成するプログラムのクラス図

もう一度，作成するプログラムのクラス図を確認しておきましょう．MainFrameクラスがメイン画面となり，ReservationControlクラスがデータベースに対する処理を行います．



作成するプログラムのメイン画面

作成するプログラムのメイン画面を以下のようにすることで話を進めます．実習においては独自の画面構成にしてもかまいません．

各ボタンをクリックすることで，利用者用サブシステムが提供する機能を実行することができます．

- ログインボタン：ログインのためのユーザIDとパスワードを入力するダイアログウィンドウ(LoginDialogクラス)が表示され，入力したユーザIDとパスワードで認証されます．
- 施設概要ボタン：施設を選ぶ選択用ボックスで施設を指定し，施設概要ボタンをクリックすると施設の説明が表示されます．これはログインしていない状態でも利用することができます．
- 空き状況確認ボタン：施設と年月日を指定して空き状況確認ボタンをクリックすると，その施設の指定した日の予約状況が表示されます．この機能もログインしていない状態で利用することができます．
- 新規予約ボタン：新規予約のための情報を入力するダイアログウィンドウ(ReservationDialogクラス)が表示され，入力した情報をもとに予約情報が新規に追加されます．追加の前に，これまでに登録されている予約情報と施設と日時の重なりがないかどうかをチェックします．
- 予約の確認ボタン：自分のこれまでの予約情報が表示されます．
- 予約のキャンセルボタン：自分のこれまでの予約情報がダイアログウィンドウ(CancelDialog)に表示され，キャンセルしたい予約を選択すると，その予約がデータベースから削除されます．

利用者用サブシステム **version0**

最初に，第9回の例題プログラムの作成時と同様に，まずは，何も表示されないウィンドウが起動するだけのプログラムを作成しましょう．作成方法の考え方も同じです．
前回作成したプロジェクトreservation_systemに，client_systemというパッケージを作成し，その中にクラスを作成していきます．最初に，ReservationControlクラス，MainFrameクラス，ReservationSystemクラスの３つのクラスを作成します．

ReservationControlクラスは新規作成で自動生成されたままにしておきます．

MainFrameクラスはawt.*とawt.event.*をインポートし，extendsを使ってFrameクラスを継承するようにします．また，implementsを使ってActionListener，WindowListener，KeyListenerを指定します．コントローラのReservationControlにメッセージを送れるようにするために，変数nControlを設けておきます．コンストラクタMainFrameでは引数としてReservationControlクラスのrcを持たせ，これを使ってrControlにコントローラのインスタンスを入れておきます．コンストラクタの最後には，WindowListenerとKeyListenerとしてthis(つまりMainFrame)を追加する処理を記述します．

ここまでできたら，左に表示された警告のアイコンをクリックします．すると，以下の図のように「実装されていないメソッドの追加」が出てきますから，これをダブルクリックしてメソッドを自動的に追加します．最後に生成されたメソッドの中のwindowClosingというメソッドの中に，System.exit(0)というプログラムを終了させる命令を記述します．

ReservationSystemクラスでは，mainメソッドの中で，ReservationControlクラスのインスタンスを生成し，それを引数としてMainFrameクラスのインスタンスを生成してmainFrameに入れます．次にMainFrameに対してsetBoundsで表示場所とサイズを指定し，setVisibleで可視化します．

以上により，単にウィンドウが表示され，右上の×をクリックすると閉じるだけのアプリケーションになるはずです．一度，第9回の授業で同じことをやっており，クラス名の違いだけ気をつければ同じようにできるはずですので，操作例や実行例の画面は付けません．

メイン画面の実装方針

次にメイン画面を実装してversion 1とします．その前に，先に示したメイン画面を実装する方針について説明しておきます．
今回のプログラムでは，メイン画面に配置するコンポーネントが多いので，レイアウトマネージャは第9回のときのFlowLayoutではなく，BorderLayoutを使用します．これを使用すると，ウィンドウの上，中央，下をそれぞれ，NORTH，CENTER，SOUTHで指定することができます．左，中央，右を，WEST，CENTER，EASTで指定することもできます．
パネルクラス(Panel)を使って，パネルにボタンやテキストフィールドなどのコンポーネントを配置し，パネルをウィンドウ(MainFrame)に配置します．具体的には，以下の図のように，３つのパネル(panelNorth，panelMid，PanelSouth)を使って，PanelNorthはその上をさらに3つのパネル(panelNorthSub1，panelNorthSub2，panelNorthSub3)を使って配置します．

panelNorthSub1には「施設予約システム」というラベルとログインボタンを配置します．panelNorthSub2には「施設」というラベルに続いて施設を選択する選択用ボックス(Choiceクラス)を配置し，施設概要を表示するための施設概要ボタンを配置します．panelNorthSub3には年月日を入力するテキストフィールドと空き状況確認ボタンを配置します．最後にこれらの３つのパネルをpanelNorthに配置します．
panelMidには，結果を表示するためのテキストフィールドを配置します．また，PanelSouthにはログインした後，利用できる機能として，新規予約ボタン，予約の確認ボタン，予約のキャンセルボタンを配置します．

メイン画面の実装

前のページで述べた方針で，利用者サブシステムのメイン画面を実装します．

FacilityChoiceクラス

まず，施設を選択するための選択用ボックスのクラスを追加します．新規作成でクラスを選択し，クラス名をFacilityChoiceクラスとします．このクラスは以下のように，Choiceクラスを継承して，コンストラクタ(FacilityChoice)の中で，本システムで利用される施設名を追加しておきます．Choiceクラスを使用するためには，awt.*をインポートしておく必要があります．

```
package client_system;

import java.awt.*;

public class ChoiceFacility extends Choice {

    ChoiceFacility(){
        add("小ホール");
        add("大会議室1");
        add("大会議室2");
        add("小会議室1");
        add("小会議室2");
        add("小会議室3");
        add("小会議室4");
        add("小会議室5");
        add("小会議室6");
    }

}
```

MainFrameクラスの変数

MainFrameクラスに以下の変数を追加しておきます．これらのうち，reservationControlはversion 0で作成しているはずです．

```
ReservationControl reservationControl;

Panel panelNorth;        //上部パネル
Panel panelNorthSub1;    //上部パネルの上
Panel panelNorthSub2;    //上部パネルの中央
Panel panelNorthSub3;    //上部パネルの下
Panel panelMid;          //中央パネル
Panel panelSouth;        //下部パネル

Button buttonLog;           // ログイン・ログアウト ボタン
Button buttonExplanation;   // 施設概要 説明ボタン
Button buttonVacancy;       //  空き状況確認
Button buttonReservation;   // 新規予約ボタン
Button buttonConfirm;       // 予約の確認
Button buttonCancel;        // 予約のキャンセルボタン

ChoiceFacility choiceFacility;      // 施設選択用選択ボックス
TextField tfYear,tfMonth,tfDay;// 年月日のテキストフィールド
TextArea textMessage;       // 結果表示用メッセージ欄
```

前のページの実装の方針で述べたパネルのための変数，ボタンのための変数，テキストフィールドの変数，結果を表示するテキストエリアの変数，施設の選択ボックスの変数です．ログインボタンをbuttonLogという変数名にし，注釈欄にログイン・ログアウトボタンと説明しているのは，同じボタンをログインする前はログインボタンとして，ログイン状態のときはログアウトボタンとして使用するためです．

MainFrameのコンストラクタ

以下にMainFrameのコンストラクタを示します．

```
public MainFrame( ReservationControl rc){

    reservationControl = rc;

    // ボタンの生成
    buttonLog = new Button(" ログイン ");
    buttonExplanation = new Button("施設概要");
    buttonVacancy = new Button("空き状況確認");
    buttonReservation = new Button("新規予約");
    buttonConfirm = new Button("予約の確認");
    buttonCancel = new Button("予約のキャンセル");

    // 設備チョイスボックスの生成
    choiceFacility = new ChoiceFacility();
    tfYear = new TextField("",4);
    tfMonth = new TextField("",2);
    tfDay = new TextField("",2);

    // 上中下のパネルを使うため，レイアウトマネージャーにBorderLayoutを設定
    setLayout( new BorderLayout());


    // 上部パネルの上パネルに　予約システム というラベルと [ログイン]ボタンを追加
    panelNorthSub1 = new Panel();
    panelNorthSub1.add(new Label("施設予約システム                "));
```

```
    panelNorthSub1.add(buttonLog);

    // 上部パネルの中央パネルに｜｜｜施設｜｜[施設名選択]チョイス　[概要説明]ボタンを追加
    panelNorthSub2 = new Panel();
    panelNorthSub2.add(new Label("施設｜｜"));
    panelNorthSub2.add( choiceFacility);
    panelNorthSub2.add(new Label("｜｜｜"));
    panelNorthSub2.add( buttonExplanation);

    // 上部パネルのしたパネルに年月日入力欄と｜空き状況確認ボタンを追加
    panelNorthSub3 = new Panel();
    panelNorthSub3.add(new Label("｜｜"));
    panelNorthSub3.add(tfYear);
    panelNorthSub3.add(new Label("年"));
    panelNorthSub3.add(tfMonth);
    panelNorthSub3.add(new Label("月"));
    panelNorthSub3.add(tfDay);
    panelNorthSub3.add(new Label("日｜"));
    panelNorthSub3.add( buttonVacancy);

    // 上部パネルに3つのパネルを追加
    panelNorth = new Panel(new BorderLayout());
    panelNorth.add(panelNorthSub1, BorderLayout.NORTH);
    panelNorth.add(panelNorthSub2, BorderLayout.CENTER);
    panelNorth.add(panelNorthSub3, BorderLayout.SOUTH);
    // メイン画面(MainFrame)に上部パネルを追加
    add(panelNorth,BorderLayout.NORTH);


    // 中央パネルにテキストメッセージ欄を設定
    panelMid = new Panel();
    textMessage = new TextArea( 20, 80);
    textMessage.setEditable(false);
    panelMid.add(textMessage);
    // メイン画面(MainFrame)に中央パネルを追加
    add( panelMid,BorderLayout.CENTER);


    // 下部パネルにボタンを設定
    panelSouth = new Panel();
    panelSouth.add(buttonReservation);
    panelSouth.add(new Label("｜｜｜"));
    panelSouth.add(buttonConfirm);
    panelSouth.add(new Label("｜｜｜"));
    panelSouth.add(buttonCancel);
    // メイン画面(MainFrame)に下部パネルを追加
    add( panelSouth,BorderLayout.SOUTH);

    //ボタンのアクションリスナの追加
    buttonLog.addActionListener(this);
    buttonExplanation.addActionListener(this);
    buttonVacancy.addActionListener(this);
    buttonReservation.addActionListener(this);
    buttonConfirm.addActionListener(this);
    buttonCancel.addActionListener(this);

    addWindowListener(this);
    addKeyListener(this);
}
```

メイン画面がある程度複雑なので，コンストラクタのリストは長くなっていますが，やっていること自体は第9回のときのプログラムとほぼ同じですから，一つ一つを丁寧に追っていけば，何をしているかはわかると思います．
Labelで引数に空白の文字列を指定しているのは，コンポーネント間に間隔をおきたいためです．また，コンポーネントを追加する際に，第2引数で位置を指定しています．BorderLayout.NORTHが上，BorderLayout.CENTERが中央，BorderLayout.SOUTHが下を示しています．

ReservationSystemクラス

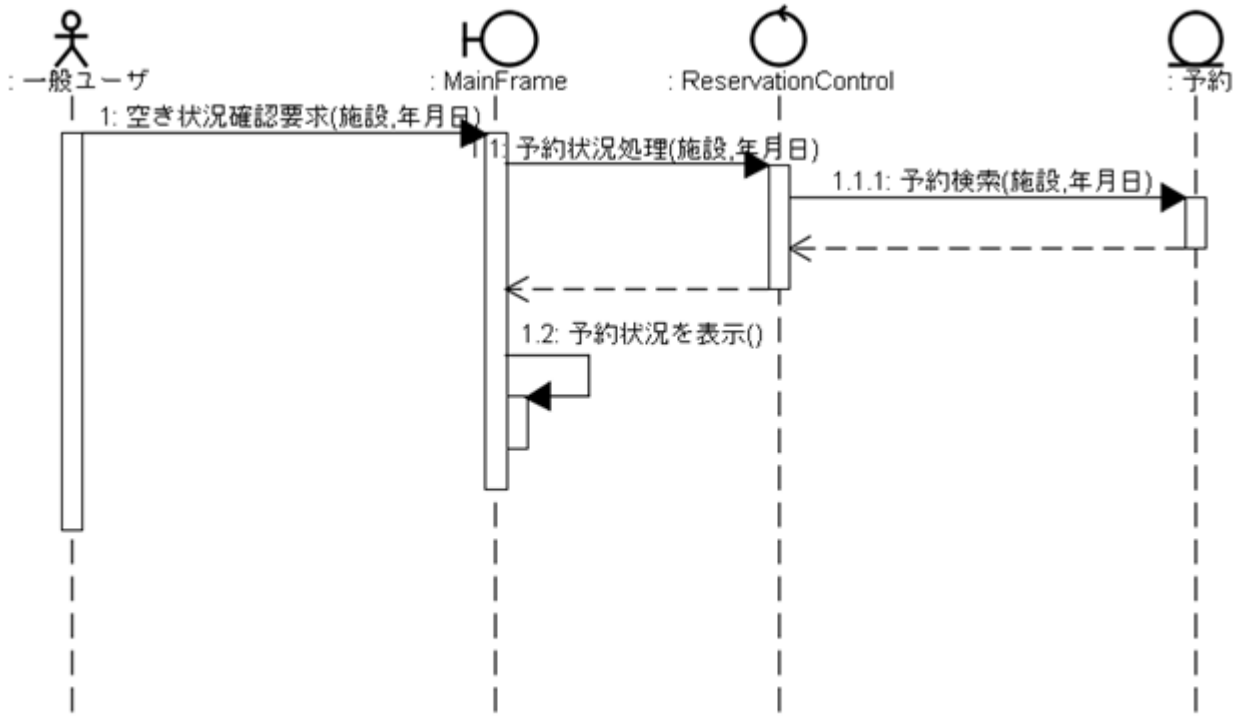
mainメソッドの中で，表示されるウィンドウのサイズを設定しているsetBoundsの呼び出しの引数を以下の程度にしておくと，先に示したメイン画面のように表示されます．必要に応じて修正しておきましょう．

```
setBounds( 5, 5, 620, 500);
```

以上ができたら，実行してメイン画面が表示されることを確認しておきましょう

空き状況確認機能の実装方針

空き状況確認ボタンをクリックしたときの処理を実装します． 空き状況確認の処理は以下のようなシーケンス図としてモデル化していました.



ユーザが施設を選択して年月日を入力した後，空き状況確認ボタンをクリックするのが「空き状況確認要求」です．これをMainFrameが受けると，予約状況の処理をResevationControlに依頼します．ReservationControlはデータベースの予約テーブルを検索し，結果を得ます．その結果をMainFrameに返し，MainFrameが予約状況をテキストエリアに表示します．
これを実装するために，ReservationControlクラスに，予約状況の処理を行うメソッドgetReservationOnを作成します．次に，MainFrameでは空き状況ボタンをクリックしたときに，getReservationOnを呼び出し，戻り値で得た文字列をテキストエリアに設定して表示します．そのため，getReservationOnでは，予約が存在しなかったときのメッセージも返すようにします.

空き状況確認機能の実装

前のページで述べた方針に従って実装しましょう．

ReservationControlクラス

コントローラであるReservationControlからSQLデータベースを操作することになるので，java.sql.*をインポートし，MySQLを操作するために使用する以下の変数を定義しておきます．
変数は第9回と作成したプログラムと同じですが，ユーザIDとパスワードが異なることに注意しましょう．

```
    //MySQLデータベース接続のための変数
    Connection sqlCon;
    Statement sqlStmt;
    String userid = "reservation_user"; //ユーザID
    String password = "pass0004";  //パスワード
```

次に，MySQLの使用準備をするためのメソッドconnectDBとデータベースの切断を行うメソッドcloseDBを追加しておきます．これら2つのメソッドのプログラムリストは第9回のプログラムと同じです．

以上の準備ができたら，施設名と予約の年月日を指定して，それらの予約状況を文字列で返すメソッドgetReservationOnを実装します．

文字列型のfacility(施設)，ryear_str(年)，rmonth_str(月)，rday_str(日)を引数として与えています．

```
    ////指定した日，施設の 空き状況 (というか予約状況)
    public String getReservationOn( String facility, String ryear_str, String rmonth_str, String rday_str){

        String res = "";

        // 年月日が数字かどうかををチェックする処理
        try {
            int ryear = Integer.parseInt( ryear_str);
            int rmonth = Integer.parseInt( rmonth_str);
            int rday = Integer.parseInt( rday_str);
        } catch(NumberFormatException e){
            res ="年月日には数字を指定してください";
            return res;
        }

        res =  facility + "  予約状況\n\n";

        // 月と日が一桁だったら，前に0をつける処理
        if (rmonth_str.length()==1) {
            rmonth_str = "0" + rmonth_str;
        }
        if ( rday_str.length()==1){
            rday_str = "0" + rday_str;
```

```
        }
        //SQLで検索するための年月日のフォーマットの文字列を作成する処理
        String rdate = ryear_str + "-" + rmonth_str + "-" + rday_str;

        //(1) MySQLを使用する準備
        connectDB();

        //(2) MySQLの操作(SELECT文の実行)
        try {
              // 予約情報を取得するクエリ
              String sql = "SELECT * FROM db_reservation.reservation WHERE date ='" + rdate + "' AND facility_name = '"+ facility +"' ORDER BY start_time;";
              // クエリーを実行して結果セットを取得
              ResultSet rs = sqlStmt.executeQuery(sql);
              // 検索結果から予約状況を作成
              boolean exist = false;
              while(rs.next()){
                    String start = rs.getString("start_time");
                    String end = rs.getString("end_time");
                    res += "    " + start + " -- " + end + "\n";
                    exist = true;
              }
              if ( !exist){ //予約が1つも存在しない場合の処理
                  res = "予約はありません";
              }
        } catch (Exception e) {
            e.printStackTrace();
        }

        //(3) MySQLへの接続切断
        closeDB();

        return res;
    }
```

プログラムリストの前半にある「年月日が整数であることをチェックし，月と日が1桁の場合には前に0をつける処理」は，第9回のプログラムにもありましたね．それに続くMySQLの操作ではSELECT文で施設名(facility)と予約日(rdate)の条件で検索し，予約の開始時刻順に並べ替えたレコードをrsに得ます．次に繰り返し処理(while文)でrsから1つずつレコードを取り出し，文字列変数resに結果を設定しています．ここでboolean型の変数existを使い，レコードが1つも存在しない場合(existがfalseの場合)は「予約はありません」という結果をresに設定します．最後にデータベースの切断を行い，resを返します．

MainFrameクラス

MainFrameのactionPerformedクラスにbuttonVacancyがクリックされたときに，上のgetReservationOnを呼び出すようなコードを追加します．

```
@Override
public void actionPerformed(ActionEvent arg0) {
    // TODO 自動生成されたメソッド・スタブ
    String result = new String();
    textMessage.setText("");
    if ( arg0.getSource() == buttonVacancy){ //// 空き状況確認ボタン
        result = reservationControl.getReservationOn( choiceFacility.getSelectedItem(),
                tfYear.getText(), tfMonth.getText(), tfDay.getText());
    }
    textMessage.setText(result);
}
```

最初に結果を得るための文字列result変数を定義しておきます．次に，結果を表示するテキストエリア(textMessage)に空の文字列を設定することでクリアしておきます．その後，クリックされたボタンが空き状況確認ボタン(buttonVacancy)だったときに，reservationControlに対してgetReservationOnを呼び出します．最初の引数は設備選択用ボックス(choiceFacility)から選択されている項目を得て，そのまま引数として渡します．同様に，年月日を表すテキストフィールドから入力されているテキストを得て，引数として渡しています．結果はresultに代入します．最後にテキストエリア(textMessage)にresultを設定します．
この後，新しく実装する機能もすべて文字列型の結果を返し，それをそのままテキストエリアに表示するようにします．

実行例

以上の実装ができたら，実行してみましょう．以下に実行例を示します．MySQLを起動していないと正しく動作しませんので，注意してください．

第11回の課題で設定した初期データにあわせて施設名と年月日を指定しないと，「予約はありません」と表示されることになります．

プログラムリスト

version 1のプログラムリストをまとめて示しておきます.

ChoiceFacilityクラス

```
package client_system;

import java.awt.*;

public class ChoiceFacility extends Choice {

        ChoiceFacility(){
                add("小ホール ");
                add("大会議室1");
                add("大会議室2");
                add("小会議室1");
                add("小会議室2");
                add("小会議室3");
                add("小会議室4");
                add("小会議室5");
                add("小会議室6");
        }

}
```

ReservationControlクラス

```
package client_system;

import java.sql.*;

public class ReservationControl {

        //MySQLデータベース接続のための変数
        Connection sqlCon;
        Statement sqlStmt;
        String sql_userid = "reservation_user"; //ユーザID
        String sql_password = "pass0004";       //パスワード

        //データベースの操作準備
        private void connectDB(){
                try{
              // ドライバクラスをロード
              Class.forName("org.gjt.mm.mysql.Driver"); // MySQLの場合

              // データベースへ接続
              String url = "jdbc:mysql://localhost?
useUnicode=true&characterEncoding=SJIS";
             sqlCon = DriverManager.getConnection(url, sql_userid, sql_password);

              // ステートメントオブジェクトを生成
              sqlStmt = sqlCon.createStatement();
                } catch (Exception e) {
                      e.printStackTrace();
            }
        }

        //データベースの切断
        private void closeDB(){
                try{
                        sqlStmt.close();
                        sqlCon.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
```

```
        ////指定した日，施設の 空き状況(というか予約状況)
        public String getReservationOn( String facility, String ryear_str, String
rmonth_str, String rday_str){

                String res = "";

                // 年月日が数字かどうかををチェックする処理
                try {
                        int ryear = Integer.parseInt( ryear_str);
                        int rmonth = Integer.parseInt( rmonth_str);
                        int rday = Integer.parseInt( rday_str);
                } catch(NumberFormatException e){
                        res ="年月日には数字を指定してください";
                        return res;
                }

                res =   facility + "  予約状況\n\n";

                // 月と日が一桁だったら，前に0をつける処理
                if (rmonth_str.length()==1) {
                        rmonth_str = "0" + rmonth_str;
                }
                if ( rday_str.length()==1){
                        rday_str = "0" + rday_str;
                }
                //SQLで検索するための年月日のフォーマットの文字列を作成する処理
                String rdate = ryear_str + "-" + rmonth_str + "-" + rday_str;

                //(1) MySQLを使用する準備
                connectDB();

                //(2) MySQLの操作(SELECT文の実行)
                try {
                      // 予約情報を取得するクエリ
                      String sql = "SELECT * FROM db_reservation.reservation
WHERE date ='" + rdate + "' AND facility_name = '"+ facility +"' ORDER BY
start_time;";
                      // クエリーを実行して結果セットを取得
                      ResultSet rs = sqlStmt.executeQuery(sql);
                      // 検索結果から予約状況を作成
                      boolean exist = false;
                      while(rs.next()){
                                String start = rs.getString("start_time");
                                String end = rs.getString("end_time");
                                res += "    " + start + " -- " + end + "\n";
                                exist = true;
                      }
                      if ( !exist){     //予約が１つも存在しない場合の処理
                          res = "予約はありません";
                      }
                } catch (Exception e) {
                        e.printStackTrace();
                }

                //(3) MySQへの接続切断
                closeDB();

                return res;
        }


}
```

MainFrameクラス

```
package client_system;

import java.awt.*;
import java.awt.event.*;


public class MainFrame extends Frame implements
ActionListener,WindowListener,KeyListener{

        ReservationControl reservationControl;

        Panel panelNorth;               //上部パネル
```

```
        Panel panelNorthSub1;           //上部パネルの上
        Panel panelNorthSub2;           //上部パネルの中央
        Panel panelNorthSub3;           //上部パネルの下
        Panel panelMid;         //中央パネル
        Panel panelSouth;               //下部パネル

        Button buttonLog;                       // ログイン ・ ログアウト ボタン
        Button buttonExplanation;               // 施設概要 説明ボタン
        Button buttonVacancy;                   //  空き状況確認
        Button buttonReservation;               // 新規予約ボタン
        Button buttonConfirm;                   // 予約の確認
        Button buttonCancel;                    // 予約のキャンセルボタン

        ChoiceFacility choiceFacility;          // 施設選択用選択ボックス
        TextField tfYear,tfMonth,tfDay;         // 年月日のテキストフィールド
        TextArea textMessage;                   // 結果表示用メッセージ欄

        public MainFrame( ReservationControl rc){

                reservationControl = rc;

                // ボタンの生成
                buttonLog = new Button(" ログイン ");
                buttonExplanation = new Button("施設概要");
                buttonVacancy = new Button("空き状況確認");
                buttonReservation = new Button("新規予約");
                buttonConfirm = new Button("予約の確認");
                buttonCancel = new Button("予約のキャンセル");

                // 設備チョイスボックスの生成
                choiceFacility = new ChoiceFacility();
                tfYear = new TextField("",4);
                tfMonth = new TextField("",2);
                tfDay = new TextField("",2);

                // 上中下のパネルを使うため，レイアウトマネージャーにBorderLayoutを設定
                setLayout( new BorderLayout());


                // 上部パネルの上パネルに　予約システム というラベルと [ログイン]ボタンを追加
                panelNorthSub1 = new Panel();
                panelNorthSub1.add(new Label("施設予約システム                  "));
                panelNorthSub1.add(buttonLog);

                // 上部パネルの中央パネルに　　施設　[施設名選択]チョイス　　[概要説明]ボタン
を追加
                panelNorthSub2 = new Panel();
                panelNorthSub2.add(new Label("施設  "));
                panelNorthSub2.add( choiceFacility);
                panelNorthSub2.add(new Label("    "));
                panelNorthSub2.add( buttonExplanation);

                // 上部パネルのしたパネルに年月日入力欄と 空き状況確認ボタンを追加
                panelNorthSub3 = new Panel();
                panelNorthSub3.add(new Label("  "));
                panelNorthSub3.add(tfYear);
                panelNorthSub3.add(new Label("年"));
                panelNorthSub3.add(tfMonth);
                panelNorthSub3.add(new Label("月"));
                panelNorthSub3.add(tfDay);
                panelNorthSub3.add(new Label("日 "));
                panelNorthSub3.add( buttonVacancy);


                // 上部パネルに３つのパネルを追加
                panelNorth = new Panel(new BorderLayout());
                panelNorth.add(panelNorthSub1, BorderLayout.NORTH);
                panelNorth.add(panelNorthSub2, BorderLayout.CENTER);
                panelNorth.add(panelNorthSub3, BorderLayout.SOUTH);
                // メイン画面(MainFrame)に上部パネルを追加
                add(panelNorth,BorderLayout.NORTH);


                // 中央パネルにテキストメッセージ欄を設定
                panelMid = new Panel();
                textMessage = new TextArea( 20, 80);
                textMessage.setEditable(false);
                panelMid.add(textMessage);
                // メイン画面(MainFrame)に中央パネルを追加
                add( panelMid,BorderLayout.CENTER);


                // 下部パネルにボタンを設定
```

```
                panelSouth = new Panel();
                panelSouth.add(buttonReservation);
                panelSouth.add(new Label("   "));
                panelSouth.add(buttonConfirm);
                panelSouth.add(new Label("   "));
                panelSouth.add(buttonCancel);
                // メイン画面(MainFrame)に下部パネルを追加
                add( panelSouth,BorderLayout.SOUTH);

                //ボタンのアクションリスナの追加
                buttonLog.addActionListener(this);
                buttonExplanation.addActionListener(this);
                buttonVacancy.addActionListener(this);
                buttonReservation.addActionListener(this);
                buttonConfirm.addActionListener(this);
                buttonCancel.addActionListener(this);

                addWindowListener(this);
                addKeyListener(this);
        }


        @Override
        public void actionPerformed(ActionEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ
                String result = new String();
                textMessage.setText("");
                if ( arg0.getSource() == buttonVacancy){ //// 空き状況確認ボタン
                        result = reservationControl.getReservationOn(
choiceFacility.getSelectedItem(),
                                                        tfYear.getText(),
tfMonth.getText(), tfDay.getText());
                }
                textMessage.setText(result);
        }

        @Override
        public void windowActivated(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void windowClosed(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void windowClosing(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ
                System.exit(0);

        }

        @Override
        public void windowDeactivated(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void windowDeiconified(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void windowIconified(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void windowOpened(WindowEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }

        @Override
        public void keyPressed(KeyEvent arg0) {
                // TODO 自動生成されたメソッド・スタブ

        }
```

```
	@Override
	public void keyReleased(KeyEvent arg0) {
		// TODO 自動生成されたメソッド・スタブ

	}

	@Override
	public void keyTyped(KeyEvent arg0) {
		// TODO 自動生成されたメソッド・スタブ

	}

}
```

ReservationSystemクラス

```
public class resevation_test {

	public static void main ( String argv[]){
		ReservationTestControl reservationControl = new
ReservationTestControl();
		ReservationTestMainFrame mainFrame = new
ReservationTestMainFrame(reservationControl);
		mainFrame.setBounds( 5,5,655,455);
		mainFrame.setVisible(true);
	}
}
```
