付録(ソースコード)

・0-00_Config

```
//---------| TellMe |---------
var obj_VersionInfo = {//バージョン
  "ver_Major": "-1",//メジャー
  "ver_Minor": "15",//マイナー
  "ver_Total": 311,//デプロイの累計
  "ver_Suffix": "Alpha",//段階
}
//----------------------------

//設定ファイル

//バージョン情報文字列の生成
var str_VersionInfo = obj_VersionInfo['ver_Major'] + "." + obj_VersionInfo['ver_Min
or'] + "." + obj_VersionInfo['ver_Total'] + " " + obj_VersionInfo['ver_Suffix'];


//++++++++++ 設定 ++++++++++
//  ○バージョン情報を常に出すかどうか
var bool_VersionOutput = false;
//++++++++++++++++++++++++++


//アクセストークン
var token_Access = "<<認証コードのため、伏せる>>";
//アクセストークンを使える形に整形
var token_Bearer = "Bearer " + token_Access;
```

```
//+++++日時の取得+++++
var date_Now = new Date();

var obj_TimeNow = {//現在の日時オブジェクト
  "Year": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "yy")),//年
  "Month": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "MM")),//月
  "Date": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "dd")),//日
  "DayCode": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "u")),//曜日
  "Hour": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "HH")),//時間
  "Minute": parseFloat(Utilities.formatDate(date_Now, "Asia/Tokyo", "mm"))//分
}

var arr_DayJp = ["日", "月", "火", "水", "木", "金", "土"];//曜日：日本語


//+++++ログ用オブジェクト+++++
var obj_PushLog = {
  "date_Recv": obj_TimeNow['Year'].toString() + "/" + obj_TimeNow['Month'].toString
() + "/" + obj_TimeNow['Date'].toString(),
  "time_Recv": obj_TimeNow['Hour'].toString() + ":" + obj_TimeNow['Minute'].toStrin
g(),
  "state_Process": "処理中",
  "event_type": "<UNDEFINED>",
  "message_type": "<UNDEFINED>",
  "text_Received": "<UNDEFINED>",
  "actcode_Whole": "<UNDEFINED>",
  "actcode_Head": "<UNDEFINED>",
  "actcode_Content": "<UNDEFINED>",
  "data_Send": "<UNDEFINED>",
  "errorcode": "NONE"
```

```
}
//++++++++++++++++++++++++++

//+++++シート設定+++++
//◇データベース用スプレッドシートを取得
var sprsht_Main = SpreadsheetApp.getActiveSpreadsheet();


//+++++シート本体の名前+++++
//◇定型句用シートの取得
var sheet_Phrase = sprsht_Main.getSheetByName('定型句');
//  ○定型句 - 定型句コードの列
var col_PhraseCode = 1;
//  ○定型句 - 定型句の内容の列
var col_PhraseValue = 2;
//  ○定型句 - 開始行
var row_PhraseStart = 3;


//◇教員データシートの取得
var sheet_Teacher = sprsht_Main.getSheetByName('教員');
//  ○教員 - 開始列
var col_TeacherStart = 3;
//  ○教員 - 教員名の行
var row_TeacherName = 2;
//  ○教員 - 研修日の行
var row_TeacherTrainDay = 3;
//  ○教員 - 研究室 - 建物の行
var row_TeacherLabBuilding = 4;
//  ○教員 - 研究室 - 階層の行
```

```
var row_TeacherLabFloor = 5;
//  〇教員 - 研究室 - 部屋番号の行
var row_TeacherLabRoom = 6;
//  〇教員 - 担当科目の授業コードの開始行
var row_TeacherClassCodeStart = 7;


//◇選択肢一覧シートの取得
var sheet_Option = sprsht_Main.getSheetByName('選択肢一覧');
//  〇選択肢 - 入力の列
var col_OptionChoice = 1;
//  〇選択肢 - アクションコードの列
var col_OptionAction = 2;
//  〇選択肢 - 開始行
var row_OptionStart = 3;


//◇メニューの json 記録用シートの取得
var sheet_MenuJson = sprsht_Main.getSheetByName('メニューjson');
//  〇選択肢 - メニューコードの行
var row_MenuCode = 2;
//  〇選択肢 - json の行
var row_MenuJson = 3;
//  〇選択肢 - 開始列
var col_MenuStart = 2;


//◇授業一覧シートの取得
var sheet_ClassInfo = sprsht_Main.getSheetByName('授業一覧');
//  〇授業一覧 - 開始行
var row_ClassStart = 3;
```

```
//  〇授業一覧 - 授業コードの列
var col_ClassCode = 1;
//  〇授業一覧 - 科目名の列
var col_ClassName = 2;
//  〇授業一覧 - 教室の列
var col_ClassRoom = 3;
//  〇授業一覧 - 教員の列
var col_ClassTeacher = 4;


//◇教室・施設名シートの取得
var sheet_Room = sprsht_Main.getSheetByName('教室・施設名');
//  〇教室・施設名 - 開始行
var row_RoomStart = 3;
//  〇教室・施設名 - 教室・施設名の列
var col_RoomName = 1;
//  〇教室・施設名 - 建物の列
var col_RoomBuilding = 2;
//  〇教室・施設名 - 階層の列
var col_RoomFloor = 3;


//◇建物座標シートの取得
var sheet_Coordinate = sprsht_Main.getSheetByName('建物座標');
//  〇建物座標 - 開始行
var row_CoordinateStart = 3;
//  〇建物座標 - 建物名の列
var col_CoordinateBuilding = 1;
//  〇建物座標 - 緯度の列
var col_CoordinateLatitude = 2;
```

```
//  〇建物座標 - 経度の列
var col_CoordinateLongitude = 3;



//◇ログシートの取得
var sheet_Log  = sprsht_Main.getSheetByName('ログ');
//  〇ログ - 統計のデータ列
var col_LogStatData = 2;
//  〇ログ - 統計のポスト回数の行
var row_LogStatPost = 3;
//  〇ログ - 統計の正常終了の行
var row_LogStatOk = 4;
//  〇ログ - 統計のエラーの行
var row_LogStatError = 5;
//  〇ログ - 処理ログの開始行
var row_LogProcessStart = 10;
//  〇ログ - 処理ログの終了状態の列
var col_LogProcessEnd = 1;
//  〇ログ - 処理ログの日付の列
var col_LogProcessTsDate = 2;
//  〇ログ - 処理ログの時刻の列
var col_LogProcessTsTime = 3;
//  〇ログ - 処理ログの質問の列
var col_LogProcessQuestion = 4;
//  〇ログ - 処理ログのアクションコードの列
var col_LogProcessActCode = 5;
//  〇ログ - 処理ログの送信の列
var col_LogProcessSend = 6;
//  〇ログ - 処理ログのエラーコードの列
var col_LogProcessErrorCode = 7;
```

```
//  〇ログ - 処理ログのバージョンの列
var col_LogProcessVersion = 8;
//++++++++++++++++++++




//var sheet_receivedjson = sprsht_Main.getSheetByName('received_json');//テスト用
//var arr_DayEng = ["Sun", "Mon", "Tue", "Wed", "The", "Fri", "Sat"];//曜日：英語短縮

```

・1-00_Main

```
//Tell Me 本体ファイル


function doPost(e) {//LINE から何かが来たら、実行
  var json_Received = e.postData.getDataAsString();// 受信したデータを文字列として取得
  var token_Reply = JSON.parse(json_Received).events[0].replyToken;//リプライトークンを取り出す
  var text_Received = JSON.parse(json_Received).events[0].message.text;//メッセージの本体を取り出す
  var eventtype_Received = JSON.parse(json_Received).events[0].type;
  var messtype_Received = JSON.parse(json_Received).events[0].message.type;//メッセージの種別を取り出す
  var userId_Posted = JSON.parse(json_Received).events[0].source.userId;//送信したユーザーの ID を取得する


  var test_viewjson = e.postData.getDataAsString();


  sheet_receivedjson.getRange(2, 1).setValue(test_viewjson);//テスト用


  obj_PushLog['message_type'] = messtype_Received;


  obj_PushLog['event_type'] = eventtype_Received;


  var url_Reply = "https://api.line.me/v2/bot/message/reply";//リプライ用 URL


  try {
    if (messtype_Received != "text") {//もし、受信したメッセージが、文字ではなかったら...
```

```
      obj_PushLog['text_Received'] = "テキスト以外：" + messtype_Received;

      var obj_ActionCode = {//アクションコード用オブジェクトを、以下の内容で作る
        "Code_Head": "show",//定型文を見せるヘッドコード
        "Code_Content": "NotText"//テキスト以外を受信したときの定型文を見せるコンテントコード
      }

    } else {//もし、そうでなければ...

      obj_PushLog['text_Received'] = text_Received;//受信ログを入力

      var obj_ActionCode = func_Decode(text_Received);//テキストから、アクションコードを取り出す
    }

    obj_PushLog['actcode_Whole'] = obj_ActionCode['Code_Head'] + "_" + obj_ActionCode['Code_Content'];
    obj_PushLog['actcode_Head'] = obj_ActionCode['Code_Head'];
    obj_PushLog['actcode_Content'] = obj_ActionCode['Code_Content'];

    /*
    obj_PushLog['state_Process'] = "エラー";
    var json_Return = func_MakeErrorJson(token_Reply, "TEST-STOP");//エラーメッセージ用 json を作る
    return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
    */
```

```
    switch (obj_ActionCode['Code_Head']){//もし、アクションコードヘッドが...
      case "show"://「show」のとき(定型文を見せるコード)は
        var text_Reply = func_FetchPhrase(obj_ActionCode['Code_Content']);//定型文を取り出す

        if (text_Reply == "ERR_LostConfCode"){//もし、定型文が取り出せていなかったら...

          obj_PushLog['state_Process'] = "エラー";
          var json_Return = func_MakeErrorJson(token_Reply, "LOST_PHRASE");//エラーメッセージ用 json を作る
          return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
          throw new Error("OK_End");
        }

        if (text_Reply == "ERR_Test") {//もし、テスト用エラーコードが帰ってきたら...

          obj_PushLog['state_Process'] = "エラーテスト：終了";
          var json_Return = func_MakeErrorJson(token_Reply, "TEST");//エラーメッセージ用 json を作る
          return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
          throw new Error("OK_End");
        }

        obj_PushLog['data_Send'] = "定型文：" + obj_ActionCode['Code_Content'];
        var json_Reply = func_MakeTextJson(text_Reply);//テキスト返信用 json を作る
        break;//switch 文を抜ける

      case "menu"://「menu」のとき(メニューを見せるコード)は
```

```
        var str_MenuJson = func_FetchMenuJson(obj_ActionCode['Code_Content']);//メニュー用 JSON をテキストとして取得

        if (str_MenuJson == "ERR_LostMenuCode") {//もし、メニュー用 JSON が取り出せていなかったら...

          obj_PushLog['state_Process'] = "エラー";
          var json_Return = func_MakeErrorJson(token_Reply, "LOST_MENUJSON");//エラーメッセージ用 json を作る
          return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
          throw new Error("OK_End");
        }
        //sheet_MenuJson.getRange(2, 5).setValue("If OK");//テスト用
        //Utilities.sleep(2000);

        var json_Reply = JSON.parse(str_MenuJson);//テキストを JSON に変換する
        //sheet_MenuJson.getRange(2, 5).setValue("parse OK");//テスト用
        //Utilities.sleep(2000);

        obj_PushLog['data_Send'] = "メニュー：" + obj_ActionCode['Code_Content'];//返信ログを入力
        obj_PushLog['state_Process'] = "終了";
        break;//switch 文を抜ける

      case "TeacherLocation"://「TeacherLocation」のとき(教員の現在地を示すコード)は

       var text_Reply = func_FetchTeacherLocation(obj_ActionCode['Code_Content']);//現在位置を取り出す
```

```
        if (text_Reply == "ERR_LostTeacherInfo") {//もし、先生の現在地が取り出せなかったら...

          obj_PushLog['state_Process'] = "エラー";
          var json_Return = func_MakeErrorJson(token_Reply, "LOST_TEACHERINFO");//エラーメッセージ用 json を作る
          return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
          throw new Error("OK_End");
        }

        obj_PushLog['data_Send'] = "現在地：" + obj_ActionCode['Code_Content'];
        var json_Reply = func_MakeTextJson(text_Reply);//テキスト返信用 json を作る

      break;//switch 文を抜ける

      case "RoomLocation"://「RoomLocation」のとき(教室・施設の場所を示すコード)は

        var json_Reply = func_FetchRoomLocation(obj_ActionCode['Code_Content']);//位置情報を取り出す

        if (json_Reply == "ERR_LostRoomLocation" || json_Reply == "ERR_LostBuildingCoordinate") {//もし、位置情報がが取り出せなかったら...

          obj_PushLog['state_Process'] = "エラー";
          var json_Return = func_MakeErrorJson(token_Reply, json_Reply);//エラーメッセージ用 json を作る
          return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
          throw new Error("OK_End");
```

```
        }

        obj_PushLog['data_Send'] = "位置：" + obj_ActionCode['Code_Content'];//返信
ログを入力
        obj_PushLog['state_Process'] = "終了";

        break;//switch 文を抜ける


      default://どれとも合わないときは
        obj_PushLog['state_Process'] = "エラー";
        var json_Return =  func_MakeErrorJson(token_Reply, "ERROR_MAIN-SWITCH");//
エラーメッセージ用 json を作る
        return UrlFetchApp.fetch(url_Reply, json_Return);//返信する
    }

    //sheet_MenuJson.getRange(2, 5).setValue("Break OK");//テスト用
    //Utilities.sleep(2000);

    obj_PushLog['state_Process'] = "終了";
    var json_Send = func_MakePayload(token_Reply, json_Reply);//返信用の json を作る
    //引数：HTTP ヘッダーjson, リプライトークン, 返すテキスト

    //sheet_MenuJson.getRange(2, 5).setValue("func OK");//テスト用
    //Utilities.sleep(2000);

    return UrlFetchApp.fetch(url_Reply, json_Send);//返信する
    //引数：返信用 URL, 返信用 json
    throw new Error("OK_End");
```

```
  } catch (e) {



    var debug = json_Reply;

    throw new Error("OK_End");


    sheet_Room.getRange(3, 5).setValue(debug);//テスト用

  }
}
```

・2-00_MakePayload

```
//TellMe 関数：00_MakePayload
//返信用 json 生成関数

function func_MakePayload(token_Reply, json_Reply) {//引数：リプライトークン，メッセージ本体の json

  var header_HTTP = {
    "Content-Type": "application/json",
    "Authorization": token_Bearer
  };

  var str_PushLog = "Date : " + obj_PushLog['date_Recv'] + "\n" +
    "Time : " + obj_PushLog['time_Recv'] + "\n" +
    "State : " + obj_PushLog['state_Process'] + "\n" +
    "Event Type : " + obj_PushLog['event_type'] + "\n" +
    "Mess. Type : " + obj_PushLog['message_type'] + "\n" +
    "Received : " + obj_PushLog['text_Received'] + "\n" +
    "A-Code : " + obj_PushLog['actcode_Whole'] + "\n" +
    "A-Code_Head : " + obj_PushLog['actcode_Head'] + "\n" +
    "A-Code_Content : " + obj_PushLog['actcode_Content'] + "\n" +
    "Send : " + obj_PushLog['data_Send'] + "\n" +
    "errorcode:" + obj_PushLog['errorcode'];

  var str_Meta = "Ver. " + str_VersionInfo + "\n" + "----------obj_PushLog---------" + "\n" + str_PushLog;

  if (bool_VersionOutput == true) { //常にバージョン情報を出力する設定のときは...
    var json_Payload = {//送信するデータ本体の json
```

```
      "replyToken": token_Reply, //リプライトークン
      "messages": [
        json_Reply,
        {
          "type": "text",//バージョン情報も付ける
          "text": str_Meta
        },
      ]
    };
  } else {//そうでなければ...
    var json_Payload = {//送信するデータ本体の json だけを、送る
      "replyToken": token_Reply, //リプライトークン
      "messages": [json_Reply],
    };
  }



  var json_Send = {//送信する json を作る
    "method": "POST",
    "headers": header_HTTP,//HTTP ヘッダーjson[0]
    "payload": JSON.stringify(json_Payload)//送信するデータ本体
  };

  sheet_receivedjson.getRange(2, 2).setValue(JSON.stringify(json_Payload));//テスト
用

  func_PushLog();//ログを記録
```

```
  return json_Send;//返信用 json を返り値として返す
}
```

・2-01_MakeTextJson

```
//TellMe 関数：01_MakeTextJson
//テキストを返信するときの json を生成する関数

function func_MakeTextJson(text_Reply){
  //返信するテキスト

  var json_Return = {
    "type": "text",
    "text": text_Reply
  };

  return json_Return;//json を返す
}
```

・2-02_MakeErrorJson

```
//TellMe 関数：02_MakeErrorJson
//エラー発生時用関数

function func_MakeErrorJson(token_Reply, str_ErrorCode) {
  //引数：リプライトークン

  //メッセージを作る
  var text_Reply = "申し訳ありません。" + "\n" +
    "エラーが発生しましたので、終了します。" + "\n" +
    "お手数ですが、IT センターまでお問い合わせください。" + "\n" +
    "エラーコード：" + str_ErrorCode;

  obj_PushLog['data_Send'] = "定型文：エラー";//返信の概要を代入
  obj_PushLog['errorcode'] = str_ErrorCode;//エラーコードを代入

  var json_text = func_MakeTextJson(text_Reply);//テキスト json を作る
  var json_Send = func_MakePayload(token_Reply, json_text);//返信用の json を作る

  return json_Send;//json を返す
}
```

・2-03_Decode

```
//TellMe 関数：03_Decode
//受信したテキストから、アクションコードを返す関数

function func_Decode(text_Received){

  //受信したテキスト

  var obj_ActionCode = {//アクションコード用オブジェクトを、用意する
    "Code_Head": "",
    "Code_Content": ""
  }

  //var i = row_OptionStart;
  //対応する選択肢が見つかるまで、ループ

  var suffix_TeacherName = "先生";//先生の名前が入力されたことを判定するための接尾辞

  // もし、入力された文字列の末尾が「先生」であれば...
  if ((text_Received.lastIndexOf(suffix_TeacherName) + suffix_TeacherName.length == text_Received.length) && (suffix_TeacherName.length <= text_Received.length)) {

    var str_TeacherNameNonSufx = text_Received.replace("先生", "");//接尾辞を抜いた教員名

    var obj_ActionCode = {//アクションコード用オブジェクトを、以下の内容で作る
      "Code_Head": "TeacherLocation",//先生の場所を見せるヘッドコード
      "Code_Content": str_TeacherNameNonSufx//コンテントコード：先生の名前(敬称略)
```

```
    }

    return obj_ActionCode;//アクションコード用オブジェクトを返す
  }

  var suffix_RoomName = "の場所";//教室などの名前が入力されたことを判定するための接尾辞

  // もし、入力された文字列の末尾が「の場所」であれば...
  if ((text_Received.lastIndexOf(suffix_RoomName) + suffix_RoomName.length === text_Received.length) && (suffix_RoomName.length <= text_Received.length)) {

    var str_RoomNameNonSufx = text_Received.replace("の場所", "");//接尾辞を抜いた教室などの名

    var obj_ActionCode = {//アクションコード用オブジェクトを、以下の内容で作る
      "Code_Head": "RoomLocation",//教室などの場所を見せるヘッドコード
      "Code_Content": str_RoomNameNonSufx//コンテントコード：教室などの名前
    }

    return obj_ActionCode;//アクションコード用オブジェクトを返す
  }


  //選択肢一覧の入力列の最終行の番号を取得
  var row_OptionLast = sheet_Option.getRange(sheet_Option.getMaxRows(), col_OptionChoice).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();
```

```
  /*
  while (sheet_Option.getRange(i, col_OptionChoice).getValue() != text_Received) {
    i++;

    if (i == (1 + row_OptionLast)) {//もし、対応する選択肢が見つからなかったら

      var obj_ActionCode = {//アクションコード用オブジェクトを、以下の内容で作る
        "Code_Head": "show",//定型文を見せるヘッドコード
        "Code_Content": "NotMatch"//対応する選択肢が無いときの定型文を見せるコンテントコード
      }

      return obj_ActionCode;//アクションコード用オブジェクトを返す
    }
  }
  */

  var str_ActionCode = "";//アクションコード用変数

  for (var i = row_OptionStart;i <= row_OptionLast ; i++) {//選択肢の列を探し終わるまで、ループ

    if (sheet_Option.getRange(i, col_OptionChoice).getValue() == text_Received){//もし、一致する選択肢が見つかったら...

      str_ActionCode = sheet_Option.getRange(i, col_OptionAction).getValue();//アクションコードを取り出す
      var arr_ActionCode = str_ActionCode.split('_');//アンダーバーで区切ったアクションコードを、作業用配列に代入
```

```
      obj_ActionCode['Code_Head'] = arr_ActionCode[0];//アクションコードのヘッドコードを代入
      obj_ActionCode['Code_Content'] = arr_ActionCode[1];//アクションコードのコンテントコードを代入

      return obj_ActionCode;//アクションコード用オブジェクトを返す
    }
  }

  var obj_ActionCode = {//アクションコード用オブジェクトを、以下の内容で作る
    "Code_Head": "show",//定型文を見せるヘッドコード
    "Code_Content": "NotMatch"//対応する選択肢が無いときの定型文を見せるコンテントコード
  }

  // var str_ActionCode = sheet_Option.getRange(i, col_OptionAction).getValue();//アクションコードを取り出す
  // var arr_ActionCode = str_ActionCode.split('_');//アンダーバーで区切ったアクションコードを、作業用配列に代入
  // obj_ActionCode['Code_Head'] = arr_ActionCode[0];//アクションコードのヘッドコードを代入
  // obj_ActionCode['Code_Content'] = arr_ActionCode[1];//アクションコードのコンテントコードを代入

  return obj_ActionCode;//アクションコード用オブジェクトを返す

}
```

・2-04_FetchPhrase

```
//TellMe 関数：04_FetchPhrase
//定型文を取得する関数

function func_FetchPhrase(actcode_Content){
  //アクションコードのコンテントコード

  switch(actcode_Content){//アクションのコンテントコードが...
    case "ShowVersion"://「ShowVersion」のとき(バージョンを表示するコード)は
      obj_PushLog['data_Send'] = "バージョン情報";//返信ログを入力
      return "Ver. " + str_VersionInfo;//バージョン情報を返す

    case "ShowTest"://「ShowTest」のとき(テストメッセージを表示するコード)は
      obj_PushLog['data_Send'] = "テストテキスト";//返信ログを入力
      return "Hello, World!";//テストメッセージを返す

    case "ShowError"://「ShowError」のとき(エラーメッセージを表示するコード)は
      return "ERR_Test";//エラーテスト用コードを返す

    default://どれとも合わないときは
      //設定シートの設定コード列の最終行を取得
      var row_ConfCodeLast = sheet_Phrase.getRange(sheet_Phrase.getMaxRows(), col_P
hraseCode).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();
      //sheet_Phrase.getRange((row_ConfCodeLast + 1), 4).setValue("^EOR");//テスト
用

      //var i = row_PhraseStart;
      //定型文の設定コードが見つかったら、ループを抜ける
```

```
      for (var i = row_PhraseStart; i <= row_ConfCodeLast; i++){//定型文コードの列
を探しきるまで、ループ

        if (sheet_Phrase.getRange(i, col_PhraseCode).getValue() == actcode_Content)
{//もし、一致するコードがあれば...

          return sheet_Phrase.getRange(i, col_PhraseValue).getValue();//定型文を返
す
        }
      }

      /*
      while (sheet_Phrase.getRange(i, col_PhraseCode).getValue() != actcode_Content
) {
        //sheet_Phrase.getRange(i, 4).setValue("＜READ");//テスト用
        i++;
        //Utilities.sleep(1500);

        if (i == (1 + row_ConfCodeLast)) {//もし、定型文の設定コードが見つからなかっ
たら
          obj_PushLog['state_Process'] = "エラー";
          return "ERR_LostConfCode";//エラーメッセージを返す
        }
      }
      */

      obj_PushLog['state_Process'] = "エラー";//for 文の中でリターンが起こらなかった
(= 一致するコードが無かった)ときは、
      return "ERR_LostConfCode";//エラーメッセージを返す
```

```
  }
}
```

・2-05_FetchMenuJson

```
//TellMe 関数：05_FetchMenuJson
//メニュー用 JSON を取得する関数

function func_FetchMenuJson(actcode_Content) {
  //アクションコードのコンテントコード

  //メニューJSON シートのメニューコード行の最終列を取得
  var col_MenuCodeLast = sheet_MenuJson.getRange(row_MenuCode, sheet_MenuJson.getMa
xColumns()).getNextDataCell(SpreadsheetApp.Direction.PREVIOUS).getColumn();
 // sheet_MenuJson.getRange(2, (1 + col_MenuCodeLast)).setValue("＜EOC");//テスト用

 /*
 var i = col_MenuStart;
  //メニューコードが見つかったら、ループを抜ける
  while (sheet_MenuJson.getRange(row_MenuCode, i).getValue() != actcode_Content) {
   // sheet_MenuJson.getRange(2, (1 + col_MenuCodeLast)).setValue("＜EOC READ：
" + i);//テスト用
    i++;
    //Utilities.sleep(1500);

    if (i == (1 + col_MenuCodeLast)) {//もし、メニューコードが見つからなかったら
      obj_PushLog['state_Process'] = "エラー";
     // sheet_MenuJson.getRange(2, (1 + col_MenuCodeLast)).setValue("＜
EOC READERR" + i);//テスト用
```

```
      return "ERR_LostMenuCode";//エラーメッセージを返す
    }
  }
  */

  for (var i = col_MenuStart; i <= col_MenuCodeLast; i++){//メニューコード行を探しきるまで、ループ

    if (actcode_Content == sheet_MenuJson.getRange(row_MenuCode, i).getValue()){//もし、同じコードがあったら...

      return sheet_MenuJson.getRange(row_MenuJson, i).getValue();//文字列形式で返す
    }
  }

  obj_PushLog['state_Process'] = "エラー";//もし、for 文の中のリターンで返らず、ここまで来たら(対応するコードが発見できなかったとき)
  return "ERR_LostMenuCode";//エラーメッセージを返す
}
```

・2-06_FetchTeacherLocation

```
//TellMe 関数：06_FetchTeacherLocation
//教員の現在地を取得する関数

function func_FetchTeacherLocation(str_TeacherName){
   try{
    //var str_TESTRTN = "JAMP_OK";

  //探している先生のフルネーム

  //教員シートの教員名行の最終列を取得
  var col_TeacherLast = sheet_Teacher.getRange(row_TeacherName, sheet_Teacher.getMa
xColumns()).getNextDataCell(SpreadsheetApp.Direction.PREVIOUS).getColumn();
    //str_TESTRTN = "TEACHERLAST_OK";
  //sheet_MenuJson.getRange(row_TeacherName, (1 + col_TeacherLast)).setValue("＜
EOC");//テスト用

  /*
  var i = col_TeacherStart;
  //先生の名前が見つかったら、ループを抜ける
  while (sheet_Teacher.getRange(row_TeacherName, i).getValue(str_TeacherName) != st
r_TeacherName) {
    //heet_MenuJson.getRange(row_TeacherName, (1 + col_TeacherLast)).setValue("＜
EOC COL_READ：" + i);//テスト用
    i++;
    //Utilities.sleep(1500);

    if (i == (1 + col_TeacherLast)) {//もし、先生の名前が見つからなかったら
      obj_PushLog['state_Process'] = "エラー";
```

```
      sheet_MenuJson.getRange(row_TeacherName, (1 + col_TeacherLast)).setValue("<
EOC READERR");//テスト用
      return "ERR_LostTeacherInfo";//エラーメッセージを返す
    }
  }
  */

  var col_ThisTeacher = -1;//対象の先生の列番号

  for (var i = col_TeacherStart; i <= col_TeacherLast; i++ ){//教員名の行を探しきる
まで、ループ

    if (sheet_Teacher.getRange(row_TeacherName, i).getValue() == str_TeacherName){/
/もし、一致する教員名が見つかったときは...

      var col_ThisTeacher = i;//対象の先生の列番号を代入
    }
  }

    //str_TESTRTN = "FOR1_OK";

  if (col_ThisTeacher == -1) {//もし、先生の名前が見つからなかったら

    obj_PushLog['state_Process'] = "エラー";
    return "ERR_LostTeacherInfo";//エラーメッセージを返す
  }

    //str_TESTRTN = "IF_OK";
```

```
  if (obj_TimeNow['DayCode'] == 0 || obj_TimeNow['DayCode'] == 6){//もし、今日が土日
なら...
    return "今日は土日なので、先生は学校にいません。";//土日なのでいないことを伝える
メッセージを返す
  }


    //str_TESTRTN = "SS_OK";


  if (sheet_Teacher.getRange(row_TeacherTrainDay, col_ThisTeacher).getValue() == ar
r_DayJp[obj_TimeNow['DayCode']]) {//もし、今日がその先生の研修日であれば...
    return "今日、" + str_TeacherName + "先生は研修日なので、学校にいません。";//研
修日でいないことを伝えるメッセージを返す
  }


    //str_TESTRTN = "TD_OK";


  var obj_TeacherLabLocation = {//先生の研究室オブジェクト
    "Building": sheet_Teacher.getRange(row_TeacherLabBuilding, col_ThisTeacher).get
Value(),//建物
    "Floor": sheet_Teacher.getRange(row_TeacherLabFloor, col_ThisTeacher).getValue(
),//階層
    "Room": ""//部屋
  }


    //str_TESTRTN = "DEF OBJ_OK";


    var pat_NumDetect = /^\d*$/;//数値検知用の正規表現パターン
    var str_Lab = sheet_Teacher.getRange(row_TeacherLabRoom, col_ThisTeacher).getVa
lue();//部屋番号の変数
```

```
    //str_TESTRTN = "get lab_OK" + "\n" + str_Lab;

    if (pat_NumDetect.test(str_Lab) == true) {//もし、部屋番号が、数字だけであれば...

      obj_TeacherLabLocation['Room'] = str_Lab + "号室";//後ろに「号室」を付ける

  } else {//そうでなければ

      obj_TeacherLabLocation['Room'] = str_Lab;//後ろに何も付けない
  }

    //str_TESTRTN = "num check_OK";

  var arr_ThisTeacherClassCodeToday = func_FetchTodayClassCode(col_ThisTeacher);//
その先生の今日の授業のコードを配列として取得する

    //str_TESTRTN = "Fetch Classcode_OK" + "\n" + arr_ThisTeacherClassCodeToday[0]
+ "\n" + arr_ThisTeacherClassCodeToday[1] + "\n" + arr_ThisTeacherClassCodeToday[2]
 + "\n" + arr_ThisTeacherClassCodeToday[3] + "\n" + arr_ThisTeacherClassCodeToday[4
];
    //throw new error("TEST");

  var obj_DefineTeacherState = {//先生の状態定義オブジェクト
    //"状態コード": "返す文字列"
    "home": "home",//学校に先生がいない
    "class": "class",//授業中
    "Prep": "prep",//準備中
    "lab": "lab"//研究室にいる
```

```
  }

  var obj_TeacherState = func_FetchTeacherState(obj_DefineTeacherState, arr_ThisTea
cherClassCodeToday);//時間から先生の状態を取り出す

  switch (obj_TeacherState['state']){//先生の状態で分岐

    case "ERR"://エラーコードを受け取ったとき

      obj_PushLog['state_Process'] = "エラー";
    return "ERR_LostTeacherInfo";//エラーメッセージを返す

    case obj_DefineTeacherState['home']://学校に先生がいないとき

      var str_Return = str_TeacherName + "先生は、学校にいらっしゃらないかもしれませ
ん。" + "\n" + //学校にいないかもしれないことを伝えるメッセージを作る
        "先生の研究室は、" + obj_TeacherLabLocation['Building'] + "の
" + obj_TeacherLabLocation['Floor'] + "階にある
「" + obj_TeacherLabLocation['Room'] + "」です。";
    return str_Return;//メッセージを返す

    case obj_DefineTeacherState['Prep']://準備中のとき

      var code_ClassFetchInfo = parseFloat(arr_ThisTeacherClassCodeToday[obj_Teache
rState['period'] - 1]);//情報を取得する授業のコード

      var obj_ClassInfo = func_FetchClassInfo(code_ClassFetchInfo);//授業の情報を取
得する
```

```
      //str_TESTRTN = "Fetch_OK : " + obj_ClassInfo['name'];
      //throw new error("TEST");

      if (obj_ClassInfo['code'] == "ERR") {//もし、授業の情報が取得できていなかったら...

        obj_PushLog['state_Process'] = "エラー";
        return "ERR_LostTeacherInfo";//エラーメッセージを返す
      }

      var str_Return = str_TeacherName + "先生は授業の準備中かもしれません。" + "\n" + "教室は、「" + obj_ClassInfo['room'] + "」です。";
      return str_Return;//メッセージを返す

    case obj_DefineTeacherState['class']://準備中のとき

      var code_ClassFetchInfo = arr_ThisTeacherClassCodeToday[obj_TeacherState['period']- 1];//情報を取得する授業のコード
      var obj_ClassInfo = func_FetchClassInfo(code_ClassFetchInfo);//授業の情報を取得する

      if (obj_ClassInfo['code'] == "ERR") {//もし、授業の情報が取得できていなかったら...

        obj_PushLog['state_Process'] = "エラー";
        return "ERR_LostTeacherInfo";//エラーメッセージを返す
      }

      var str_Return = "現在、" + str_TeacherName + "先生は授業中です。" + "\n" +
```

```
        "教室は、「" + obj_ClassInfo['room'] + "」です。";
    return str_Return;//メッセージを返す


    case obj_DefineTeacherState['lab']://研究室にいるとき


      var str_Return = "もしかすると" + str_TeacherName + "先生は、研究室にいらっし
ゃるかもしれません。" + "\n" +//研究室の場所を案内するメッセージを作る
        "場所は、" + obj_TeacherLabLocation['Building'] + "の
" + obj_TeacherLabLocation['Floor'] + "階にある
「" + obj_TeacherLabLocation['Room'] + "」です。"
    return str_Return;//文字列を返す
  }
   } catch (e) {
     return str_TESTRTN;
     //return e.lineNumber + "\n" + e.fileName;
   }
}
```

・2-07_FetchTodayClassCode

```
//TellMe 関数：07_FetchTodayClassCode
//その教員の今日の授業コードを返す関数

function func_FetchTodayClassCode(col_ThisTeacher) {
  //その先生の列番号

  var arr_ClassCodeToday = ["NONE", "NONE", "NONE", "NONE", "NONE"];//取得した授業コードを保管する配列

  //教員シートのその先生の列の最終行を取得
  var row_TeacherLast = sheet_Teacher.getRange(sheet_Teacher.getMaxRows(), col_ThisTeacher).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();

  for (var i = row_TeacherClassCodeStart; i <= row_TeacherLast; i++) {//その先生の授業コードをすべて読み終えるまで...

    if (sheet_Teacher.getRange(i, col_ThisTeacher).getDisplayValue().substr(0, 1) == obj_TimeNow['DayCode']) {//もし、読み込んだその行が、今日の曜日の授業コードだったら

      var index_ArrayOverwright = (sheet_Teacher.getRange(i, col_ThisTeacher).getDisplayValue().substr(1, 1)) - 1;//授業コードの百のケタ(コマ数)から配列の要素番号を算出し代入
      arr_ClassCodeToday[index_ArrayOverwright] = parseInt(sheet_Teacher.getRange(i, col_ThisTeacher).getDisplayValue());//授業コードを代入

```

```
      if (sheet_Teacher.getRange(i + 1, col_ThisTeacher).getDisplayValue().substr(0
, 1) != obj_TimeNow['DayCode']) {//もし、次の行の授業コードの千のケタ(曜日)が、今日
のものと違うときは...


        break;//ループを抜ける
      }
    }
  }


  return arr_ClassCodeToday;//配列を返す
}
```

・2-08_FetchTeacherState

```
//TellMe 関数：08_FetchTeacherState
//現在の先生の状態を返す関数

function func_FetchTeacherState(obj_DefineTeacherState, arr_ThisTeacherClassCodeTod
ay) {
  //状態定義オブジェクト，今日の授業コードの配列

  var obj_TeacherState = {//先生の状態オブジェクト
    "state": "ERR",//エラー処理のための初期値
    "period": "0"
  }

  switch(obj_TimeNow['Hour']){//時間によって分岐

    case 0:
    case 1:
    case 2:
    case 3:
    case 4:
    case 5:
    case 6:
    case 7:

      obj_TeacherState['state'] = obj_DefineTeacherState['home'];//「学校にいない」
と判断(00:00 - 07:59)
    break;

    case 8:
```

```
      if (obj_TimeNow['Minute'] < 30) {//もし、8 時 30 分より前であれば(00-29)

        obj_TeacherState['state'] = obj_DefineTeacherState['home'];//「学校にいない」
と判断(00:00 - 07:59)
        break; //switch 文を抜ける
      }

      if (arr_ThisTeacherClassCodeToday[0] != "NONE"){//その先生が 1 限に授業があれば

        obj_TeacherState['state'] = obj_DefineTeacherState['Prep'];//「準備中」と判
断(00-59)
        obj_TeacherState['period'] = 1;//「次は、1 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断

      }
    break;

    case 9:

      if (arr_ThisTeacherClassCodeToday[0] != "NONE") {//その先生が 1 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(00-59)
```

```
        obj_TeacherState['period'] = 1;//「1 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断

      }

    break; //switch 文を抜ける

    case 10:

      if (obj_TimeNow['Minute'] < 30){//もし、10 時 30 分より前であれば(00-29)

        if (arr_ThisTeacherClassCodeToday[0] != "NONE") {//その先生が 1 限に授業があ
れば

          obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と
判断
          obj_TeacherState['period'] = 1;//「1 限目」と判断
          break; //switch 文を抜ける

        } else {

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
          break; //switch 文を抜ける
        }
```

```
      }

     if (obj_TimeNow['Minute'] < 40) {//もし、10 時 40 分より前であれば(30-39)

      if (arr_ThisTeacherClassCodeToday[1] != "NONE") {//その先生が 2 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['Prep'];//「準備中」と判
断
        obj_TeacherState['period'] = 2;//「次は、2 限目」と判断
        break; //switch 文を抜ける
      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

       break; //switch 文を抜ける
    }

      if (arr_ThisTeacherClassCodeToday[1] != "NONE") {//その先生が 2 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(40-59)
        obj_TeacherState['period'] = 2;//「2 限目」と判断

      } else {//なければ
```

```
        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」と判断
      }

    break; //switch 文を抜ける

    case 11:

      if (arr_ThisTeacherClassCodeToday[1] != "NONE") {//その先生が 2 限に授業があれば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判断(00-59)
        obj_TeacherState['period'] = 2;//「2 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」と判断
      }

    break; //switch 文を抜ける

    case 12:

      if (obj_TimeNow['Minute'] < 10) {//もし、12 時 10 分より前であれば(00-09)

        if (arr_ThisTeacherClassCodeToday[1] != "NONE") {//その先生が 2 限に授業があれば
```

```
          obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と
判断(40-59)
          obj_TeacherState['period'] = 2;//「2 限目」と判断

        } else {//なければ

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
        }

        break; //switch 文を抜ける
      }

      if (arr_ThisTeacherClassCodeToday[2] != "NONE") {//その先生が 3 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['Prep'];//「準備中」と判
断
        obj_TeacherState['period'] = 3;//「次は、3 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

    break; //switch 文を抜ける
```

```
    case 13:

      if (arr_ThisTeacherClassCodeToday[2] != "NONE") {//その先生が 3 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(00-59)
        obj_TeacherState['period'] = 3;//「3 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

    break; //switch 文を抜ける

    case 14:

      if (obj_TimeNow['Minute'] < 30) {//もし、14 時 30 分より前であれば(00-29)

        if (arr_ThisTeacherClassCodeToday[2] != "NONE") {//その先生が 3 限に授業があ
れば

          obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と
判断(00-59)
          obj_TeacherState['period'] = 3;//「3 限目」と判断

        } else {//なければ
```

```
          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
        }

        break; //switch 文を抜ける
      }

      if (obj_TimeNow['Minute'] < 40) {//もし、14 時 40 分より前であれば(30-39)

        if (arr_ThisTeacherClassCodeToday[3] != "NONE") {//その先生が 4 限に授業があ
れば

          obj_TeacherState['state'] = obj_DefineTeacherState['Prep'];//「準備中」と
判断
          obj_TeacherState['period'] = 4;//「次は、4 限目」と判断

        } else {//なければ

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
        }

        break; //switch 文を抜ける
      }

      if (arr_ThisTeacherClassCodeToday[3] != "NONE") {//その先生が 4 限に授業があれ
ば
```

```
        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(40-59)
        obj_TeacherState['period'] = 4;//「4 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

    break; //switch 文を抜ける

    case 15:

      if (arr_ThisTeacherClassCodeToday[3] != "NONE") {//その先生が 4 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(00-59)
        obj_TeacherState['period'] = 4;//「4 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

    break; //switch 文を抜ける
```

```
    case 16:

      if (obj_TimeNow['Minute'] < 10) {//もし、16 時 10 分より前であれば(00-09)

        if (arr_ThisTeacherClassCodeToday[3] != "NONE") {//その先生が 4 限に授業があ
れば

          obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と
判断
          obj_TeacherState['period'] = 4;//「4 限目」と判断

        } else {//なければ

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
        }

        break; //switch 文を抜ける
      }

      if (obj_TimeNow['Minute'] < 20) {//もし、16 時 20 分より前であれば(10-19)

        if (arr_ThisTeacherClassCodeToday[4] != "NONE") {//その先生が 5 限に授業があ
れば

          obj_TeacherState['state'] = obj_DefineTeacherState['Prep'];//「準備中」と
判断
          obj_TeacherState['period'] = 5;//「次は、5 限目」と判断
```

```
        } else {//なければ

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
        }

        break; //switch 文を抜ける
      }

      if (arr_ThisTeacherClassCodeToday[4] != "NONE") {//その先生が 5 限に授業があれ
ば

        obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判
断(20-59)
        obj_TeacherState['period'] = 5;//「5 限目」と判断

      } else {//なければ

        obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」
と判断
      }

    break; //switch 文を抜ける

    case 17:

      if (obj_TimeNow['Minute'] < 50) {//もし、17 時 50 分より前であれば(00-49)
```

```
        if (arr_ThisTeacherClassCodeToday[4] != "NONE") {//その先生が 5 限に授業があれば

          obj_TeacherState['state'] = obj_DefineTeacherState['class'];//「授業中」と判断(20-59)
          obj_TeacherState['period'] = 5;//「5 限目」と判断

        } else {//なければ

          obj_TeacherState['state'] = obj_DefineTeacherState['lab'];//「研究室にいる」と判断
        }

        break; //switch 文を抜ける
      }

      obj_TeacherState['state'] = obj_DefineTeacherState['home'];//「学校にいない」と判断(50-59)
    break; //switch 文を抜ける

    case 18:
    case 19:
    case 20:
    case 21:
    case 22:
    case 23:
      obj_TeacherState['state'] = obj_DefineTeacherState['home'];//「学校にいない」と判断(18:00 - 23:59)
    break;
```

```
  }


  return obj_TeacherState;//オブジェクトを返す
}
```

・2-09_FetchClassInfo

```
//TellMe 関数：09_FetchClassInfo
//授業の情報を取得する関数

function func_FetchClassInfo(code_Class){
  //授業コード

  var obj_Return= {//返すオブジェクト
    "code": "ERR",//エラー判断用コード
    "name": "",
    "room": "",
    "teacher": ""
  }

  //var arr_Return = ["ERR", "", "", ""];

  var str_CodeClass = code_Class.toString();//授業コードを文字列型に変換

  sheet_ClassInfo.getRange(3, 8).setValue("Code：" + str_CodeClass);//テスト用

  //授業コードの最後の行番号を取得
  var row_ClassLast = sheet_ClassInfo.getRange(sheet_ClassInfo.getMaxRows(), col_Cl
assCode).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();

  var arr_Class = sheet_ClassInfo.getRange(row_ClassStart, col_ClassCode, row_Class
Last - 1, col_ClassTeacher).getValues();//このシートの内容を 2 次配列として取得

  for (var i = 0; i <= (arr_Class.length - 2); i++ ) {//配列の最後の要素まで、ループ
```

```
    //var code_Read = parseFloat(sheet_ClassInfo.getRange(i, col_ClassCode).getValu
e());

    if (arr_Class[i][0] == str_CodeClass){//もし、一致する授業コードがあれば...

      obj_Return['code'] = arr_Class[i][col_ClassCode - 1];//授業コードを代入
      obj_Return['name'] = arr_Class[i][col_ClassName - 1];//科目名を代入
      obj_Return['room'] = arr_Class[i][col_ClassRoom - 1];//教室を代入
      obj_Return['teacher'] = arr_Class[i][col_ClassTeacher - 1];//教員名を代入

      // arr_Return[0] = sheet_ClassInfo.getRange(i, col_ClassCode).getDisplayValue
();//授業コードを代入
      // arr_Return[1] = sheet_ClassInfo.getRange(i, col_ClassName).getDisplayValue
();//科目名を代入
      // arr_Return[2] = sheet_ClassInfo.getRange(i, col_ClassRoom).getDisplayValue
();//教室を代入
      // arr_Return[3] = sheet_ClassInfo.getRange(i, col_ClassTeacher).getDisplayVa
lue();//教員名を代入

      sheet_ClassInfo.getRange(3, 7).setValue(obj_Return['code']);//テスト用
      sheet_ClassInfo.getRange(4, 7).setValue(obj_Return['name']);//テスト用
      sheet_ClassInfo.getRange(5, 7).setValue(obj_Return['room']);//テスト用
      sheet_ClassInfo.getRange(6, 7).setValue(obj_Return['teacher']);//テスト用

      return obj_Return;//オブジェクトを返す
    }
  }
```

```
  return obj_Return;//エラーコードがそのままの状態で、オブジェクトを返す(一致する授業コードがなかったとき)
}
```

・2-10_FetchRoomLocation

```
//TellMe 関数：10_FetchRoomLocation
//教室などの場所を返す関数

function func_FetchRoomLocation(str_RoomName) {
  //探す教室などの名前

  var json_Return = "ERR";//エラー処理用の文字列

  //教室・施設名列の最後の行番号を取得
  var row_RoomLast = sheet_Room.getRange(sheet_Room.getMaxRows(), col_RoomName).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();

  for (var i = row_RoomStart; i <= row_RoomLast; i++) {//教室などを探し切るまで、ループ

    if (str_RoomName == sheet_Room.getRange(i, col_RoomName).getValue()) {//もし、一致する名前があれば...

      var arr_BuildingCoordinate = func_FetchBuildingCoordinate(sheet_Room.getRange(i, col_RoomBuilding).getValue())//建物の座標を配列として取得

      if (arr_BuildingCoordinate[0] == "ERR") {//もし、座標の取得に失敗していたら...
```

```
        return "ERR_LostBuildingCoordinate";//エラーメッセージを返す
      }

      //建物と階を説明する文章を作る
      var str_Description = sheet_Room.getRange(i, col_RoomBuilding).getValue() + "
の、" + sheet_Room.getRange(i, col_RoomFloor).getValue() + "階です。";

      json_Return = {
        "type": "location",
        "title": str_RoomName,//教室名
        "address": str_Description,//階層の説明
        "latitude": arr_BuildingCoordinate[0],//緯度
        "longitude": arr_BuildingCoordinate[1]//経度
      };

      return json_Return;//json を返す
    }
  }

  return json_Return;//エラーコードがそのままの状態で、文字列を返す(一致する教室名が
なかったとき)
}
```

・2-11_FetchBuildingCoordinate

```
function func_FetchBuildingCoordinate(str_BuildingName) {//建物の座標を返す関数
  //対象の建物の名前

  var arr_Return = ["ERR", ""];//エラー処理用の文字列

  //建物名列の最後の行番号を取得
  var row_CoordinateLast = sheet_Coordinate.getRange(sheet_Coordinate.getMaxRows(),
 col_CoordinateBuilding).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();

  for (var i = row_CoordinateStart; i <= row_CoordinateLast; i++) {//建物を探し切る
まで、ループ

    if (str_BuildingName == sheet_Coordinate.getRange(i, col_CoordinateBuilding).ge
tValue()) {//もし、一致する名前があれば...

      arr_Return[0] = parseFloat(sheet_Coordinate.getRange(i, col_CoordinateLatitud
e).getValue());//緯度を数値として代入
      arr_Return[1] = parseFloat(sheet_Coordinate.getRange(i, col_CoordinateLongitu
de).getValue());//経度を数値として代入

      return arr_Return;//配列を返す
    }
  }

  return arr_Return;//エラーコードがそのままの状態で、文字列を返す(一致する教室名が
なかったとき)
}
```

・2-12_PushLog

```
//TellMe 関数：12_PushLog
//ログを記録する関数

function func_PushLog() {
    var num_Post = parseInt(sheet_Log.getRange(row_LogStatPost, col_LogStatData).getValue());//ポスト回数を取得
    var num_Ok = parseInt(sheet_Log.getRange(row_LogStatOk, col_LogStatData).getValue());//正常終了回数を取得
    var num_Error = parseInt(sheet_Log.getRange(row_LogStatError, col_LogStatData).getValue());//エラー回数を取得

    if (sheet_Log.getRange(sheet_Log.getMaxRows(), col_LogProcessEnd).getValue() != "") {//もし、ログシートの最後の行にデータがあったら...

        var row_LogWrite = sheet_Log.getMaxRows() + 1;//最終行より１つ行を追加したところに書き込むようにする

    } else {//そうでなければ、

        //ログシートのデータがある最終行を取得
        var row_LogLast = sheet_Log.getRange(sheet_Log.getMaxRows(), col_LogProcessEnd).getNextDataCell(SpreadsheetApp.Direction.UP).getRow();

        if (row_LogLast == (row_LogProcessStart - 2)) {//もし、最終行が見出しであれば...

            var row_LogWrite = row_LogProcessStart;//見出しの下に書き込むようにする
```

```
        } else {//そうでなければ、

            var row_LogWrite = row_LogLast + 1;//最後のログの下に書き込むようにする
        }
    }

    sheet_Log.getRange(row_LogWrite, col_LogProcessEnd).setValue(obj_PushLog['state_Process']);//終了状態を書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessTsDate).setValue(obj_PushLog['date_Recv']);//日付を書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessTsTime).setValue(obj_PushLog['time_Recv']);//日時を書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessQuestion).setValue(obj_PushLog['text_Received']);//質問を書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessActCode).setValue(obj_PushLog['actcode_Whole']);//アクションコードを書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessSend).setValue(obj_PushLog['data_Send']);//送信を書き込み
    sheet_Log.getRange(row_LogWrite, col_LogProcessVersion).setValue(str_VersionInfo);//バージョンを書き込み

    sheet_Log.getRange(row_LogStatPost, col_LogStatData).setValue(num_Post + 1);//ポスト回数を書き換え

    if (obj_PushLog['errorcode'] == "NONE") {//もし、エラーが発生してなかったら...

        sheet_Log.getRange(row_LogStatOk, col_LogStatData).setValue(num_Ok + 1);//正常終了回数を書き換え
```

```
    } else {

        sheet_Log.getRange(row_LogWrite, col_LogProcessErrorCode).setValue(obj_Push
Log['errorcode']);//エラーコードを書き込み
        sheet_Log.getRange(row_LogStatError, col_LogStatData).setValue(num_Error +
1);//エラー回数を書き換え
    }

    return; 終了
}
```