EBS-260

USER SCRIPTS (PHP) AND DIFFERENT FUNCTIONS

- Object (ComPort) allows the user to process data from the Bluetooth Module (and incoming from an external device). This allows the user to perform operations on data sent (e.g. from a Bar Code Scanner connected via Bluetooth to the printer.
- The purpose of the script is to download the sent data, process the data, and then transfer the data to the Object (ComPort). Scripts can be created by the user while maintaining a few rules:
  - Scripts should be created as .php files
  - When using a script there is no access to system commands
  - Scripts can only be used by inserting them into the "Custom Data" file
- After saving the script file in the appropriate directory (Custom Data), it can be chosen for the project as a service script object (ComPort).
- For each separator in the object (ComPort), the data that has been transferred to the separator will be passed to the script, so the script will be executed as many times as separators have been filled during data transfer.
- The following are sample scripts, along with an explanation of their individual parts. Basic Assumptions for PHP script processing data received from the Communications Port:

INPUT:

$argv [1] – Data received from the communications port are forwarded to the PHP script as a command line argument, which is shown inside the script as a variable $argv [1].

OUTPUT: All output data to be passed to the printer should be passed as a WriteResult function argument as part of the EBSFunctions Library.

EBS Functions Library:

- The printer directory CustomData contains the EBSFunctions Library.
- To load the library use: require_once ('EBSFunctions.inc')
- This function will be shown as $EBSFunc in the script.
- The library includes the following functions:
  - WriteResult ($data)
  - $data argument passed to this function will be passed back to the object of ComPort for the purpose of further processing (according to the objects settings) and for printing, for example: $EBSFunc->WriteResult ('EBS-260 HANDJET') will cause the string 'EBS-260 HANDJET' to be printed
- Excel_Search ($XLSFile, $SearchValue, $SearchInCol, $ReturnPattern)
  - This function is used to search the data in MS Excel Files
  - $XLSFile – this is the name of the MS Excel File where the search will take place; For Example:

        - BarcodeTest.xls
    - $SearchValue – this parameter specifies how the phrase is to be searched; For Example:
        - ‘EBS-260, $argv[1], etc.
    - $SearchInCol – when using this parameter, you must specify which column or columns are to be searched for a particular phrase. The columns are numbered starting from number 1. If searching several columns, the numbers should be separated by a comma; For example:
        - ‘1’, ‘1,2’, etc.
    - $ReturnPattern – this parameter defines which cells found in the file are to be returned. Without this pattern, the entire line will be returned. Numbers of columns to be returned must be preceded by a “$” sign; For example: ‘$2’ will return the contents of the second column; ‘$2 - $3’ will return the contents of the second and third columns with their separation by ‘-‘.
- Data that is found and read from the MS Excel File will be output to the printer with the Exel_Search function.
- Sample script searches for data in an Excel File might look like this:

```
<? php

ini_set(‘display_errors’,1);

require_once (‘EBSFunctions.inc’);

$result = $EBSFunc->Excel_Search (‘BarcodeTest.xls’, $argv [1], ‘1’, ‘$2’);

$EBSFunction->WriteResult ($result);

?>
```

- CSV_Search ($CSVFile, $DataSeparator, $SearchValue, $SearchInCol, $ReturnPattern)
    - This function is used to search the data in a CSV File
        - $DataSeparator – this parameter is used to specify the data separator for the CSV File
    - Other parameters are the same as listed in the Excel_Search function
- Data that is found and read from the CSV File will be output to the printer with the CSV_Search function.
- Sample script searches for data in a CSV File might look like this: <? php

```
ini_set(‘display_errors’,1);

require_once (‘EBSFunctions.inc’);

$result = $EBSFunc->CSV_Search (‘ExampleData.csv’, $argv [1], ‘1’, ‘$2’);

$EBSFunction->WriteResult ($result);

?>
```

- Retrieve data from a database
    - DB_init ($db_type, $db_host, $db_user, $db_passwd, $db_name)
    - This function is used to initiate a connection to a database
    - $db_type – database type which will be used to make a connection
    - Supported types include: MySQL, MSSQL, PostgreSQL
    - $db_host – the address of the database server
    - $db_user – the name of the database user
    - $db_passwd – the database password for the user
    - $db_name – the name of the database to be connected
    - If there is an connection problem, a ‘Connection Error” will occur and the script will end its action

    - DB_Close () – this function is used to close the database connection
    - DB_Search ($table, $where, $resultPattern) – function performs a SELECT query for the database to which you are connected.
        - $table – table in the database from which the data is retrieved
        - $where – WHERE clause to the SQL query; For Example: ‘Printer’ = “ EBS-260 “
        - $resultPattern – pattern for the return value, the column names from the database to be returned should be in “; e.g., “ ‘Name’ “ will return the contents of the Name column; “ Hello World ‘Name’ “ will return the string “Hello World” with the contents of the Name column appended to it.
    - Data that is found and read from the database will be output to the printer with the DB_Search function.

- Sample script searches for data in a database might look like this:

```
<? php

ini_set(‘display_errors’,1);

require_once (‘EBSFunctions.inc’);

$EBSFunc-
>DB_Init(‘MySQL’,’localhost’,’exampleUser’,’examplePasswd’,’exampleData’;

$result = $EBSFunc->DB_Search (‘exampleTable’, ‘Printer’ =” EBS-260 “, “ Hello World
‘Printer’”);

$EBSFunction->DB_Close ();

$EBSFunction->WriteResult ($result);

?>
```