POWER FORM USER GUIDE
Version 2025

The information in this manual is not binding and may be modified without prior notice.

Supply of the software described in this manual is subject to a user license. The software may not be used, copied or reproduced on any medium whatsoever, except in accordance with this license.

No portion of this manual may be copied, reproduced or transmitted by any means whatsoever, for purposes other than the personal use of the buyer, unless written permission is obtained from TEKLYNX Corporation SAS.

Table of Contents

CHAPTER 1

Introduction

Introduction

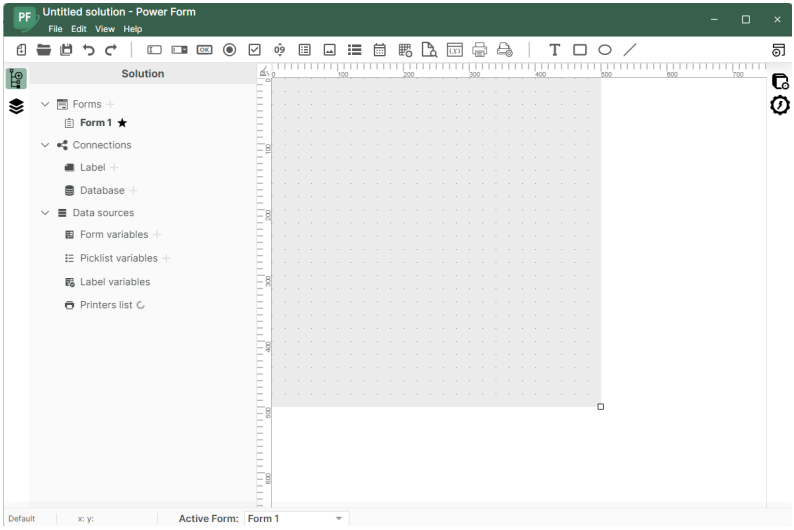
Power form is a visual designing tool that allows you to rapidly build front-end data entry and label printing applications.

It includes features such as:

- Label and printer type selection
- Print preview
- Page and printer settings
- Database connections
- Multiple screen/form application
- Pre-defined values and data control

The layout

Power form's layout is easy to use, reducing the time necessary to familiarize yourself with the software. It includes a Menu bar, a Tool bar, a Solution tree, a Properties grid, a Main window and a Status bar.



The Menu bar

The **Menu** bar contains 4 items: **File**, **Edit**, **View** and **Help**. When clicked, each item will expand showing a new list of items, to allow the user to perform a variety of functions. To access the menu items more quickly, press **Alt + the underlined letter** from your desired choice: **Alt+F** will open the File menu.

The Tool bar The **Tool** bar is usually displayed just below the Menu bar. It is made up of icons which represent the most commonly used software functions and objects to operate the form. These icons can be clicked to quickly access these functions.

Objects is made up of 15 controls and 4 shapes that can be added to the form: Text input, Combobox, Button, Radio button, CheckBox, Numeric up/down input, GroupBox, PictureBox, ListBox, DatePicker, DataTableGrid, LabelPreview, Label variables data grid, Printer Selector, Print Settings, Text, Rectangle, Ellipse and Line.

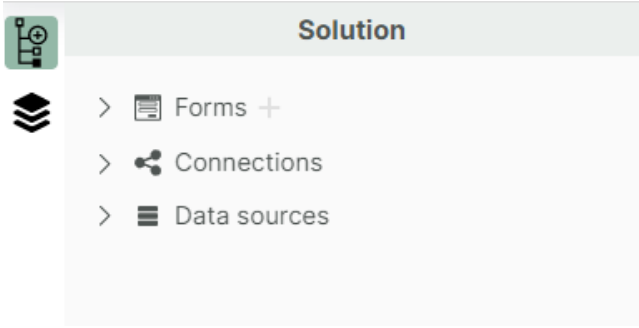
Each control serves a specific purpose. They can be linked to fixed or variable data and can be associated with actions. To insert a control into your form, simply click on the control and drag and drop it into the form.

Control	Description
Text input	Data entry field
ComboBox	Displays a drop-down list of data as defined in its properties
Button	An active control used to perform set functions/actions
Radio button	Displays a list of data as a radio group view.
CheckBox	A control used to activate/deactivate or select/deselect predefined features
Numeric up/down	Data entry field for the numeric values only
GroupBox	Non-active box placed around a group of controls for visual organization
PictureBox	A box where an image can be imported into
ListBox	Displays a list of data as defined in its properties
DatePicker	Data entry field for the date format defined in its properties
DataTableGrid	Displays the database table defined in the properties along with some basic functionality such as filters and search fields
LabelPreview	A box that will display a preview of the label attached to the form
Label variables data grid	Displays “When printed” label variables as Name, Value view
Printer Selector	Allows user to select active printer and change selected printer settings

Control	Description
Print settings	Gives user ability to print selected label, and set label quantity, label copy, page copy, and intercut parameters
Text	Non-active text used to show the name of an input field
Rectangle	Displays a shape formed as a rectangle
Ellipse	Displays a shape formed as an Ellipse
Line	Displays a shape formed as a Line

The Solution tree

The **Solution tree** is set of tabs that is displayed on the left side of the application. It is made up of sections that is used to apply general configurations to the solution.



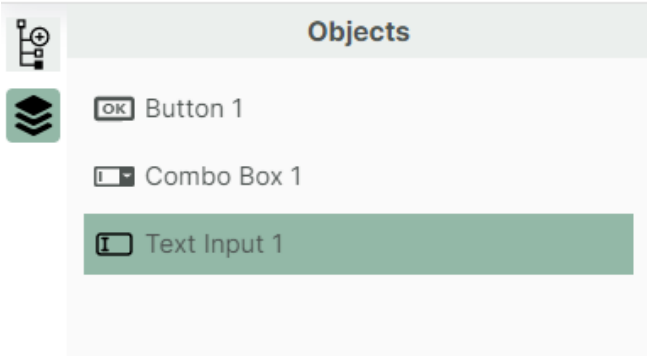
The **Solution tab** is made up of the 3 different sections that can be configured and applied to whole form solution: Forms, Connections and Data sources.

Section	Description
Forms	A list of forms that is used in the solution to represent different working areas

Connections	A list of available connections, that includes API connection, Label connection and Database connection.

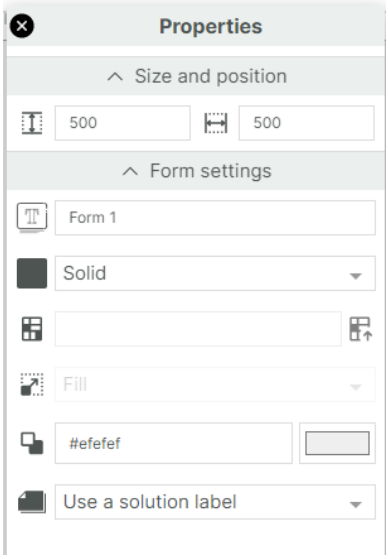
Data sources	Contains a diffenent types of variables that can be used in the solution, there a 4 types of data sources: Form variables, Picklist variables, Label variables and printers list.

The **Objects tab** is a list of objects that are placed on the active form.

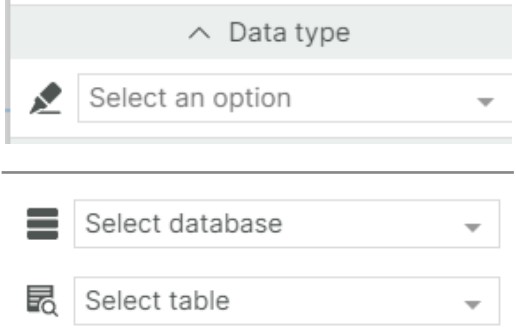


The Properties Grid

The **Properties Grid** lists all the properties for the selected control. Here you can change the characteristics of your controls such as control name, color, size, or actions. It is context-sensitive, so the list of properties will change according to the control selected. Having the grid displayed replaces the usual right-click menu command.

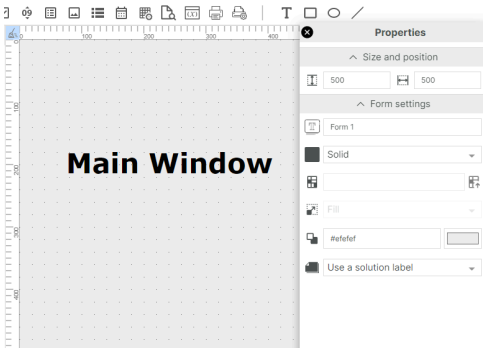


Some controls may include a Data Type section that allows users to assign a variable or link database to that control. Once assigned, control value will be binded with selected data type and any changes made to either the control or the data type will be synchronized.



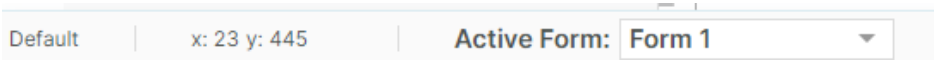
The Main Window

The **Main window**, displayed in the center, is your work area. Here, you will add controls and create the forms you will use to input data for label printing. Once the form has been created and configured, you can go to **File > Run** to run the form and see its final result.



The Status bar

The **Status bar** gives you the information about active printer, x axis and y axis location of your pointer and the active form selector dropdown.



The Basics

What are Actions?

Actions are functions that are assigned to buttons and to load/unload forms. In Power form, 9 pre-defined Actions have been included in the **Actions** menu to save you time and to eliminate the need for programming. In some cases, where specific parameters must be configured, these, too, have been predefined in the action's configuration

The predefined actions are:

Action	**Confirmation popup**
	Displays the confirmation dialog box where the user can see predefined message and buttons to accept or cancel message.
Parameters	***Title***
	Allows user to set headline title.
	Message

	Allows user to set main text of the message.
	Buttons
	Dropdown with set of different button options to be displayed in the message popup.
	Icon
	Dropdown with set of different icons to be displayed in the message popup.

Action	**Update variable value**
	Allows user to update variables.
Parameters	***Variable name***
	User can select name of the variable to be updated.
	New value type
	Dropdown of value types that can be applied to variable to be updated.
	Option
	Allows user to add new value to existing value of the selected variable.
	New value
	Field that allows user to select a new value for the selected variable.

Action	**Display form**.
	Allows the user to open a new form.
Parameters	***Form to display***

	List of forms where user can select what form to display.
	Display form behaviour
	Opens the chosen form in a new window or in the same one.
	Option
	Allows user to open previously opened form if it exists.

Action	**Exit form**.
	Allows user to close form executor window.

Action	**Set printer**
	Allows the user to print from the selected printer and override any printers that may be associated with the document.
Parameter	***Set Printer***
	Allows you to set a printer as the default printer to use when the button is clicked, or to use the printer selected from a list - if a printer selection variable is used and defined in the form properties.

Action	**Print Label**
	Allows the user to print.
Parameters	***Label count***
	Sends the printer the number of labels to be printed. If the document includes a counter, the counter increments along with the number of labels.
	Label Copy

	Lets the printer know how many identical copies of each label should be printed.
	Page Copy
	Lets the printer know how many copies of each page to print.
	Inter Cut
	Lets the printer know how many labels to print before each cut.

Action	**Read data from file**
	Allows the user to read files data and apply it value to the variable or control.
Parameters	***Choose file***
	Allows user to select file to read.
	Maximum size
	Sets maximum size of lines that will be read from the file.
	Apply value type
	Allows user to select where apply data from file it can be variable or control.
	Select variable/control
	Allows user to select to wich control data will be applied.

Action	**Open dialog**
	Allows the user to open Codesoft dialogs
Parameter	**Open dialog**

	Dropdown with list of the available dialogs to open.

Action	**Open web link**
	Allows the user to open selected web link
Parameter	**Link address**
	Input for link that will be opened to be inserted.

Action	**Execute SQL request**
	Allows the user to send SQL request to connected database for execution.
Parameter	**Database**
	Select database where SQL request will be executed
Parameter	**SQL Statement**
	Text area for SQL code

Action	**Execute Script request**
	Allows the user to run the native JavaScript code to compare values, update variables, write data to file, etc.
Parameter	**Script**
	Text area for code
Parameter	**Category**
	Dropdown with list of categories names for methods, constants, operators etc.
Parameter	**Snippets**
	List of snippets in selected category

Parameter	**Add**
	Button to add selected snipped to code text area
Parameter	**Validate**
	Button to validate the syntax of the code entered in the text area.

Action	**Refresh database**
	Allows the user to force refresh database
Parameter	**Take data from**
	A dropdown with the option 'Database' to force refresh the database, and 'From previous action result' to refresh the database using SQL action filtering.

Script action

The "Execute Script Action" allows users to bind JavaScript code to UI elements (like buttons) in a Power form application. This lets you extend the app's logic by writing scripts that respond to user interactions.

Step-by-Step Instructions

1. Select a UI Element

1. Click on the UI element you want to attach a script to (e.g. Button).
2. Tip: You can select buttons, inputs, grids, or anything that supports events.

2. Open the Events Panel

1. Click the gear icon  .
2. In the "Events" panel, find the event you want to handle — e.g. Click.
3. Press the "..." button next to the event name to configure it.

3. Add a Script Action

1. In the "Configure Event Actions" dialog that opens:
    a. Click the "Execute script action" icon  .
    b. A new action will appear in the list.

4. Write Your Script

In the "Properties" panel, enter your custom JavaScript logic.
Example:

```JavaScript
const a = 12;
const b = 32;
return b * a;
```

* This JavaScript code snippet defines two constant variables, `a` assigned the value 12 and `b` assigned the value 32. It then returns the result of multiplying `b` by `a`.

5. Validate & Apply

1. Click "Validate" to check your script for syntax errors.
2. Once successful, press "Apply" to save the action.

Using Code Snippets

The scripting interface offers a list of predefined snippets organized by category. These snippets help insert commonly used expressions, constants, structures, and functions without needing to write them from scratch.

To use a snippet:

1. Select a category from the dropdown.
2. Choose a snippet from the second dropdown.
3. Click “Add” to insert it into the script editor.

All Available Snippets

Snippet	Description
Math	
Math.sqrt(number)	Returns the square root of a number
Math.abs(number)	Returns the absolute value
Math.cos(number)	Returns the cosine of a number (radians)
Math.sin(number)	Returns the sine of a number (radians)
Math.ceil(number)	Rounds up to the nearest integer
Math.floor(number)	Rounds down to the nearest integer
Math.round(number)	Rounds to the nearest integer
Math.random()	Returns a random number between 0 and 1
Math.pow(5, 2)	Raises 5 to the power of 2
Math.max(a, b)	Returns the maximum of two values
Math.min(a, b)	Returns the minimum of two values
Math.log(x)	Natural logarithm (base e) of x
Math.exp(x)	Returns e^x
JSON	
JSON.stringify(object)	Converts an object to a JSON string
JSON.parse(jsonString)	Parses a JSON string into an object

Variables	
let x = value;	Declares a variable with block scope
const PI = 3.14;	Declares a constant
let result;	Declares an uninitialized variable
var total = 0;	Declares a function-scoped variable
Constants	
Number.MAX_VALUE	Maximum numeric value
Number.MIN_VALUE	Smallest positive numeric value
undefined	Primitive indicating no value
null	Intentional absence of any object value
NaN	Not-a-Number value
Math.PI	The ratio of circumference to diameter
Math.E	Euler's number
Math.LN2	Natural log of 2
Math.LN10	Natural log of 10
Math.LOG2E	Base 2 log of e
Math.LOG10E	Base 10 log of e
Math.SQRT1_2	Square root of 0.5

Math.SQRT2	Square root of 2
Logical	
a > b	Greater than comparison
a < b	Less than comparison
a >= b	Greater than or equal
a <= b	Less than or equal
a && b	Logical AND
a || b	Logical OR
!a	Logical NOT
a != b	Inequality comparison
a !== b	Strict inequality comparison
TRUE	Boolean true
FALSE	Boolean false
Operators	
a + b	Addition
a - b	Subtraction
a * b	Multiplication
a / b	Division

a % b	Modulo
a ** b	Exponentiation
a += b	Add and assign
a -= b	Subtract and assign
a *= b	Multiply and assign
a /= b	Divide and assign
a == b	Equality
a === b	Strict equality
++a	Pre-increment
--a	Pre-decrement
Statements	
if (condition) { }	Conditional execution
else { }	Executes if condition is false
else if (condition) { }	Checks another condition
for (let i = 0; i < n; i++) { }	For loop
while (condition) { }	While loop
do { } while (condition);	Do-while loop
switch (expression) { case: break;}	Switch-case conditional

try { } catch (e) { }	Try-catch block
return value;	Returns a value
break;	Breaks out of loop
continue;	Skips to next loop iteration
Array	
array.push(item)	Adds item to end of array
array.pop()	Removes last item from array
array.map(x => x)	Transforms each array element
array.filter(x => x > 0)	Filters elements based on condition
array.find(x => x === 0)	Finds first match
array.reduce(...)	Reduces array to a single value
array.includes(value)	Checks if value exists in array
array.slice(start, end)	Returns a shallow copy of array
array.indexOf(value)	Returns index of value in array
Object	
Object.keys(obj)	Returns keys of object
Object.values(obj)	Returns values of object
Object.entries(obj)	Returns key-value pairs

Object.assign({}, source)	Copies values into a new object
obj.hasOwnProperty(key)	Checks if object has property
String	
str.toUpperCase()	Converts to uppercase
str.toLowerCase()	Converts to lowercase
str.includes("text")	Checks for substring
str.replace("", "")	Replaces text
str.split(",")	Splits string into array
str.trim()	Removes whitespace
str.slice(start, end)	Extracts part of string
Number	
Number(value)	Converts to number
num.toFixed(2)	Formats number to fixed decimals
num.toString()	Converts number to string
Boolean	
Boolean(value)	Converts to boolean
!value	Negation
!!value	Double negation (truthy check)

MapSet	
const map = new Map();	Creates new map
map.set("key", value);	Sets key-value pair
map.get("key");	Gets value by key
map.has("key");	Checks key existence
map.delete("key");	Deletes key
const set = new Set();	Creates new set
set.add(value);	Adds value to set
set.has(value);	Checks value existence
set.delete(value);	Removes value from set
DateTime	
new Date("")	Creates date from string
dateVar.toISOString()	Converts to ISO string
dateVar.getTime()	Gets time in ms
Date.now()	Gets current timestamp
Timers	
setTimeout(() => {}, 1000);	Delays function execution
setInterval(() => {}, 1000);	Repeats function execution

clearTimeout(timeoutId);	Clears timeout
clearInterval(intervalId);	Clears interval
FileSystem	
fs.readFileSync("", "utf8")	Reads file content
fs.writeFileSync("", "")	Writes to file
fs.appendFileSync("", "")	Appends to file
const fd = fs.openSync("", "rw")	Opens file
fs.fsyncSync(fd);	Flushes file content
fs.closeSync(fd);	Closes file
fs.existsSync("")	Checks file existence
fs.readdirSync("")	Reads directory content
Functions	
function myFunction() { }	Declares a function
const add = () => null;	Declares an arrow function
myFunction();	Calls a function

Troubleshooting

Note: console.log() is not available in this scripting environment. To identify errors, review the error logs located at *%APPDATA%/TekLynx/web-forms/logs*

Issue	Solution
Script not running	Check if you clicked Apply and bound it to the right event
Validation fails	Review syntax, ensure all variables are declared

Tips

1. Use *return* to pass result back to the form system.
2. Combine multiple lines of logic — all JavaScript features are supported.

Working with Form Variables

You can directly update form variables within your script by assigning a new value to them using their name. For example, if a form variable is named candy_amount, you can set its value like this:

```javascript
candy_amount = 12;
```

The script engine will automatically recognize this assignment and apply the value to the corresponding form variable in the data source after script execution is complete.

You can also use the result of a return statement from one script action in a subsequent action by referencing the Previous action result in the next action's configuration.

France +33 (0) 562 601 080
Germany +49 (0) 2103 2526 0
Singapore +65 6908 0960
United States +1 (414) 837 4800

www.teklynx.com