Date : 15-Feb-22

PDF Generator (jsPDF)

version: 1.0.1

# **Overview**

The PDF Generator (jsPDF) component can generate a PDF file with content from HTML or a Responsive Web/ Desktop form in Iris. Under the hood, the component uses the [jsPDF library](https://github.com/MrRio/jsPDF), which is available under the [MIT License](https://opensource.org/licenses/MIT).

# **Getting Started**

## **Prerequisites**

Before you start using the PDF Generator (jsPDF) component, ensure you have the following:

* [HCL Foundry](https://manage.hclvoltmx.com/)
* Volt MX Iris

## **Platforms Supported**

### Mobile

1. iOS
2. Android

### Tablets

### PWA

## **Importing the Component**

## You can import the Forge components only into the apps that are of the Reference Architecture type.

## **To import the PDF Generator (jsPDF) component, do the following:**

## Open your app project in Volt MX Iris.

1. In the Project Explorer, click the **Templates** tab.

Graphical user interface, text, application

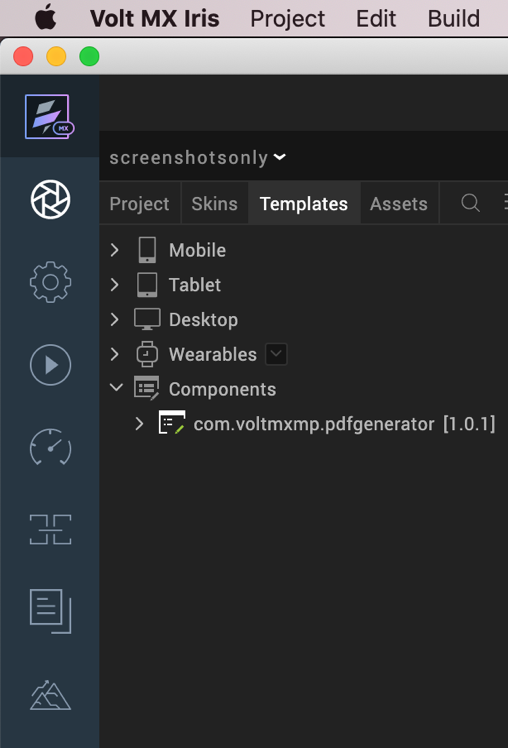
Description automatically generated

3. Right-click **Components**, and then select **Import Component**. The **Import Component** dialog box appears.

Graphical user interface, text, application, Teams

Description automatically generated

4. Click **Browse** to navigate to the location of the component, select the component, and then click **Import**. The component and its associated widgets and modules are added to your project.



Once you have imported a component to your project, you can easily add the component to a form. For more information, refer [Add a Component to a Form](https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Iris/iris_user_guide/Content/C_UsingComponents.html#add-a-component-to-a-form).

# **3. References**

## **A. Dynamic Usage**

You can also add a PDF Generator (jsPDF) component dynamically. To do so:

1. In the **Project Explorer**, on the **Projects** tab, click **Controllers** section to access the respective **Form Controller**. Create a method and implement the code snippet similar to the sample code mentioned below.

In the code snippet, you can edit the properties of the component as per your requirement. For more information, see Setting Properties.

/\* Creating a component's Object \*/

const pdfGenerator = new com.voltmxmp.pdfgenerator({

"clipBounds": true,

"height": "0dp",

"id": "pdfGenerator",

"isVisible": true,

"left": "0dp",

"top": "0dp",

"width": "0dp",

"zIndex": 1

}, {}, {});

pdfGenerator.fileName = "<pdf file name>";

pdfGenerator.orientation = "<pdf orientation>";

pdfGenerator.format = "<format>";

pdfGenerator.compress = <compress>;

pdfGenerator.unit = "<unit>";

pdfGenerator.leftprop = <left>;

pdfGenerator.topprop = <top>;

pdfGenerator.widthprop = <width>;

pdfGenerator.bottomprop = <bottom>;

this.view.add(pdfGenerator);

1. **Save** the file.

## **B. Properties**

The properties provided on the **Component** tab allow you to customize the UI elements in the **PDF Generator (jsPDF)** component. You can set the properties directly on the **Component** tab or by writing JavaScript.

**1. PDF File Name**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the name that you want to set for the generated PDF. |
| **Syntax:** | fileName |
| **Type:** | String |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.fileName = "PDF One"; |

**2. Orientation (orientation)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the orientation that you want to set for the generated PDF. |
| **Syntax:** | orientation |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * portrait (p) * landscape (l) |
| **Example:** | this.view.componentID.orientation = "portrait"; |

**3. Format (format)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the format that you want to set for the pages of the generated PDF. |
| **Syntax:** | format |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | |  |  |  |  | | --- | --- | --- | --- | | * a0 * a1 * a2 * a3 * a4 * a5 * a6 * a7 * a8 * a9 * a10 | * b0 * b1 * b2 * b3 * b4 * b5 * b6 * b7 * b8 * b9 * b10 | * c0 * c1 * c2 * c3 * c4 * c5 * c6 * c7 * c8 * c9 * c10 | * dl * letter * government-letter * legal * junior-legal * ledger * tabloid * credit-card | |
| **Default Value:** | a4 |
| **Example:** | this.view.componentID.format = "a4"; |

**4. Compress(****compress)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies whether you want to compress the generated PDF. |
| **Syntax:** | compress |
| **Type:** | Boolean |
| **Read/Write:** | Write |
| **Default Value:** | False |
| **Example:** | this.view.componentID.compress = false; |

**5. Unit(unit)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the measurement unit for the page margins. |
| **Syntax:** | unit |
| **Type:** | * List Selector * String |
| **Read/Write:** | Write |
| **Values:** | * pt * mm * cm * m * in * px |
| **Default Value:** | pt |
| **Remarks:** | The component uses the **unit** measurement for the following properties.   * left * top * width * bottom |
| **Example:** | this.view.componentID.unit = "pt"; |

**6. Left(left)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the distance of the left edge of the content from the left edge of the page. |
| **Syntax:** | leftprop |
| **Type:** | Integer |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.leftprop = 10; |

**7. Top(top)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the distance of the top edge of the content from the top edge of the page. |
| **Syntax:** | topprop |
| **Type:** | Integer |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.topprop = 10; |

**8. Width(width)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the width of the content of the page. |
| **Syntax:** | widthprop |
| **Type:** | Integer |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.widthprop = 400; |

**9. Bottom(bottom)**

|  |  |
| --- | --- |
| **Category:** | Custom |
| **Description:** | Specifies the distance of the bottom edge of the content from the bottom edge of the page. |
| **Syntax:** | bottomprop |
| **Type:** | Integer |
| **Read/Write:** | Write |
| **Example:** | this.view.componentID.bottomprop = 10; |

## **C. Events**

-- None of the events are exposed.

## **D. APIs**

The following APIs pertain to the PDF Generator (jsPDF) component.

**1. [generatePdfUsingHtmlString](javascript:void(0);)**

|  |  |
| --- | --- |
| **Description:** | Generates a PDF file from the specified HTML string. |
| **Syntax:** | generatePdfUsingHtmlString(htmlString) |
| **Parameters:** | *htmlString [String]*: Contains the HTML content to generate the PDF. |
| **Return Value:** | None |
| **Platform Availability:** | * Android * iOS * Desktop Web |
| **Example:** | var htmlString = '<div id="main">'+  '<div>'+  'Sample Html'+  '</div>'+  '</div>';  this.view.componentID.generatePdfUsingHtmlString(htmlString); |

**2. g**[**eneratePdfUsingDom**](javascript:void(0);)

|  |  |
| --- | --- |
| **Description:** | Generates a PDF file from the specified Volt MX element. |
| **Syntax:** | generatePdfUsingDom(formFlexName) |
| **Parameters:** | *formFlexName [String]*: The name of FlexContainer that you want to use to generate the PDF. |
| **Return Value:** | None |
| **Platform Availability:** | * Desktop Web |
| **Remarks:** | The name of the element is in the format **FormName\_FlexName**.  Therefore, if the name of your form is **frmHome**, and the name of your FlexContainer is **flxContainer1**, then the Volt MX element name is **frmHome\_flxContainer1**. |
| **Example:** | var formFlexName = "frmHome\_flxContainer1";  this.view.componentID.generatePdfUsingDom(formFlexName); |
|  |  |

**Note:**  
If you copy the code from any example, make sure that you replace componentID with the name of your component.

# **4. Revision History**

App version 1.0.1

## **A. Known Issues**

## No issues

## **B. Limitations**

* The component does not support generation of images. Make sure that your Volt MX Iris element or [HTML string](https://docs.kony.com/marketplace/PDFGenerator/Content/Reference.htm#generatePdfUsingHtmlString) does not contain images.
* For iOS, the pdf file is generated in the data directory of Volt MX App, so it can not be downloaded directly.