# dpa-ElectionsLive

## ReadMe file for integrating and customizing the new widgets

Dear colleagues,

This document is designed to help you customize the dpa ElectionsLive applications to suit your needs. Using the widgets for the 2025 federal election (**de-2025**) as an example, we'll walk you through the process of integrating and customizing the widgets in a few simple steps. If you have any further questions, please feel free to contact us:

Support: +49 (0)40 4113 31000
service@dpa-info.com

**Document versions**

| Topic | Comment | Date |
|---|---|---|
| Initialization | | 06.10.2025 |
| Additional URL parameters | | 11.12.2025 |
| Update use of CSS-variables (Kap. 4.3) | | 06.01.2026 |
| Parameter *partyOrder* new (Kap. 4.1) | | 14.01.2026 |
| Parameter *showFooterSource* new (Kap. 4.1) | | 04.02.2026 |
| Parameter *footerLinkURL* new (Kap. 4.1) | | 11.02.2026 |
| Parameter *infoButtonMode* new (Kap. 4.1) | | 19.02.2026 |

| | | |
|---|---|---|
| Parameter *showFinalOnly* new (Kap. 4.1) | | 03.03.2026 |
| Parameter *allowedSourceIds* new (Kap. 4.1) | | 24.04.2026 |
| Update “Changing Labels” (Kap. 4.2.1) | | 24.04.2026 |
| New Chapter “Test Data" (Kap. 5) | | 21.08.2026 |

# Table of Contents

**Table of Contents** **3**
**1. General Information** **4**
**2. Creating a custom layout directory (S3 push only)** **5**
**3. Integration of the application in an iframe** **7**
3.1 Browser support 7
3.2 Internal server communication (S3 push only) 7
3.3 Cross-server communication 8
**4. Custom settings** **10**
4.1 Widget Options 10
4.2 config.json 17
4.2.1 Changing Labels 18
4.2.2 Parties 18
4.3 Configuring Your Own Layout 19
4.3.1 Changing the header image 20
**5. Test Data** **21**
**6. Logging** **23**
**7. Accessibility** **24**
**8. GDPR** **24**

# 1. General Information

In 2026, there will be several political elections, which we will cover with polls in the period leading up to the elections and with results on election night. For clarity, we will describe how the election apps can be integrated and customized using the 2025 federal election as an example (**de-2025**).

Some of the procedures described in these chapters apply only to S3 push delivery. If your applications are hosted by us, you can already configure many settings via your setup page. For others, the information on the following pages will certainly be helpful.

| **Election** | **ID** |
|---|---|
| German Federal Election | de-2025 |
| State Election in Baden-Württemberg | de_bw-2026 |
| State Election in Rhineland-Palatinate | de_rp-2026 |
| State Election in Saxony-Anhalt | de_st-2026 |
| State Election in Mecklenburg-Western Pomerania | de_mv-2026 |
| Berlin House of Representatives Election | de_be-2026 |

You can find the dates for the elections and our releases in our Help Center:
https://dpaq.de/RszGAtN

# 2. Creating a custom layout directory (S3 push only)

We will upload the ***de-2025*** directory to your S3 bucket. This directory contains additional subdirectories for various data types, such as results data, images, and text files (.json, .js, .html, .jpg, .png, .svg). Please ensure that all corresponding MIME types are configured on your server.

In the ***de-2025*** directory on your S3 bucket, you will find a customization directory as a ZIP file: ***custom_de-2025.zip***. Before implementing your layout changes, please unzip this file. Please create this directory on your publication server under the name custom. It must not be renamed and must be located **within** the ***de-2025*** delivery directory. This is important so that the application can find your customized style.css and config.json files. If you wish to store your customization directory elsewhere, you must pass the absolute paths to the files to the application via parameters (see Chapter 4.1 for details). Afterwards, you should have the following folder structure on your web server:

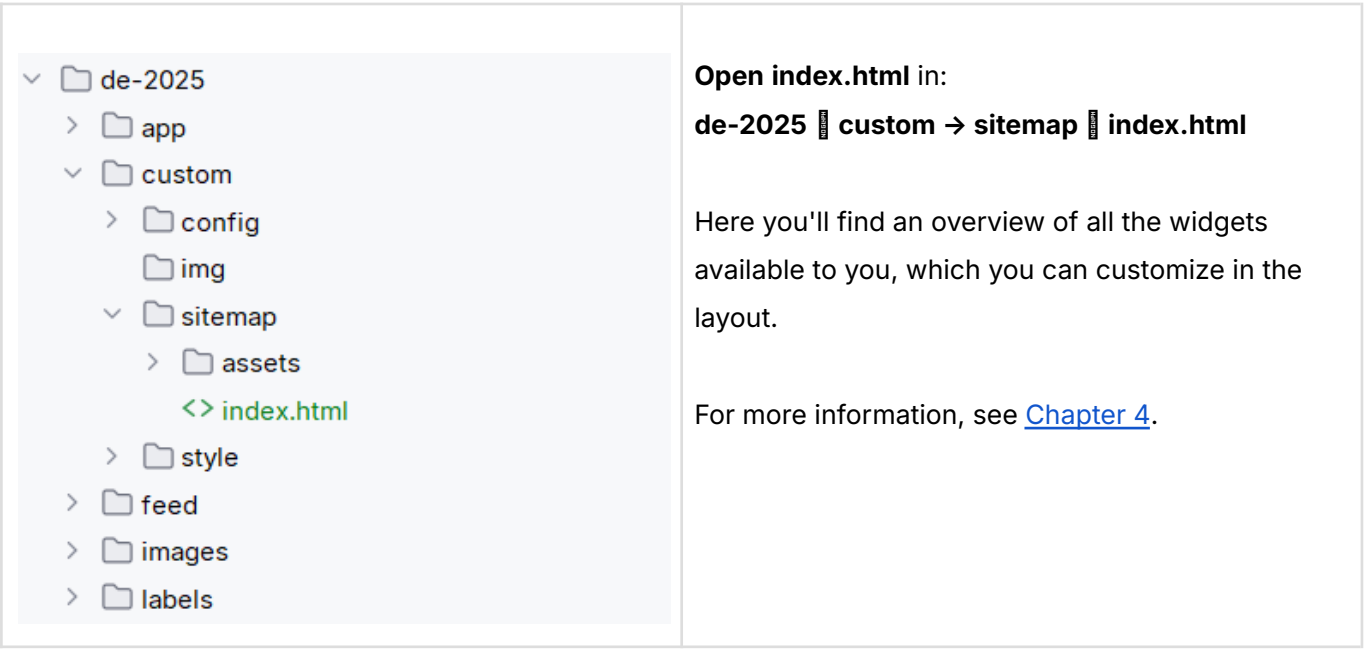



*Figure 1: Directory structure of the customization directory*

Below, we explain the implementation of the ***de-2025*** service (Federal Parliamentary Election)

| Directory structure on the web server | Subdirectory | Content |
|---|---|---|
| de-2025/**custom**<br>*(Name **cannot** be chosen freely)* | | custom layout directory |
| | config | Here you will find a **config.json** file that you can customize, and a **config.example.json** file, which is provided for **reference only**. |
| | sitemap | Index file for the widget overview. Here you can find the file path to the widget. |
| | style | Here you'll find your style.css file, where you can make CSS adjustments. All of our CSS variables are already included there, commented out. |
| de-2025<br>*(Name **cannot** be chosen freely)* | | **Standard directory** |
| | app | JavaScript, CSS, and HTML files, as well as images and graphic files contained within widgets |
| | feed | All results feeds. (JSON) |
| | images | Candidate photos, party logos, election logo (JPG, PNG, SVG) |
| | labels | Label files for various languages (JSON) |
| | **custom** | → Your customization directory, see above |

**Note**: Please make all customizations and changes only in your individual layout directory, ***de-2025/custom***, as this directory will never be overwritten by our delivery processes, even if synchronization is required. This ensures that the customizations you have made remain unchanged.

# 3. Integration of the application in an iframe

If you are a **hosting customer**, you can generate the corresponding embed code on your **setup page**. You only need to follow the steps in 3.3 if you cannot or do not want to use our embed code with dpa.js.

As an **S3 Push customer**, you can view the available widgets via **de-2025 ▮ custom → sitemap ▮ index.html** if you have created the directory structure shown above on your web server.
Each widget has its own HTML page. There is no longer a parent index.html file that uses a parameter to control which widget should be called.

The **height of the widgets** depends on their content and the user's screen width. To ensure that the embedding iframes adjust their height accordingly, we present the different methods in sections 3.2 and 3.3.
In the following examples, the "Distribution of Votes" widget is embedded.

## 3.1 Browser support

We have optimized our HTML applications for the following browsers: Chrome version 109 and later, Firefox version 118 and later, Microsoft Edge version 114 and later, and Safari version 17 and later. As of December 2020, we no longer support Microsoft Internet Explorer.

## 3.2 Internal server communication (S3 push only)

If the application is located in the same subdomain as the HTML page embedding it, the widget can automatically adjust the height of the iframe.
To do this, please follow these steps: In the sitemap (de-2025 ▮ custom → sitemap ▮ index.html), click on the desired widget and copy the URL from the address bar.
In your iframe code, set any name in **id**. Pass this ID to the widget via the ***iframeId*** parameter. You also need the ***scope*** parameter, in which you pass the domain name of the embedding page to the widget. You can do this in "src" using URL variables (see example):

```
<iframe id='dpa_frame_de-2025' src='[Your path]/de-2025/app/election/bars.html?eventId=de-2025&appLocation=push&iframeId=dpa_frame_de-2025&scope=https://yourSubDomain.yourDomain.com' scrolling="no"  width='100%'...></iframe>
```

The widget performs an internal check to verify that the domains match before attempting to access the parent window. This prevents the browser from aborting the load due to a violation of the "same-origin

policy.” If everything is correct, the height of the iframe is automatically adjusted whenever the height of the content changes.
Despite the dynamic height adjustment, scroll bars may still appear, for example, when opening the “Graph Information” window. To prevent this, please use one of these options if you do not have a general solution:

1. Add a style attribute to the iframe:
   ***style="overflow: hidden;"***
2. Add the attribute `**scrolling=“no”**` to the iframe.
   We are aware that this attribute is considered deprecated, but it is still supported by most browsers and can be useful in certain scenarios.

## 3.3 Cross-server communication

If your application is hosted on a different server than the HTML page where the application is called via an iframe, or if you are a hosting customer and do not use our embed code with dpa.js, you will need to take additional steps.

The reason for this is that the application is not allowed to access the parent window if it is not located in the same subdomain (“Same-Origin Policy”). To resolve this issue, you can embed the widgets using a static iframe and use a custom script to adjust the height.

To do this, embed the widget using a simple iframe as described in 3.2 and pass the ID of your iframe to the widget:

```
<iframe id='dpa_frame_de-2025' src='[Your path]/de-2025/app/election/bars.html?eventId=de-2025&appLocation=push&iframeId=dpa_frame_de-2025' scrolling="no" width='100%'...></iframe>
```

After loading and whenever the content changes, the widget sends an object via ***postMessage*** that contains its height, the corresponding iframe ID, and the sentinel “dpa”. You can then intercept this on your page and process it accordingly, for example like this:

```
<script type='text/javascript'>
window.addEventListener('message',function(_evt) {
        if (_evt.data.sentinel === 'dpa' && _evt.data.type === 'embed-size' && _evt.data.height > 0) {
            document.getElementById(_evt.data.embed).style.height = `${_evt.data.height}px`;
        }
})
```

```
</script>
```

We recommend querying by “type,” since we are also sending an object containing logging information (see Chapter 5).

# 4. Custom settings

You can customize your graphics according to your own preferences and ideas. You can use variables to modify certain aspects of the application's behavior, and use CSS to adapt colors, fonts, and layout to your corporate identity. We have already prepared the most important classes for customization in **style.css**. As a Push customer, you can find this file in your unzipped customization directory under ***custom/style/***; as a hosting customer, you can download it from your setup page under "**Colors/Fonts**".

## 4.1 Widget Options

By setting variables, you can customize the applications to suit your needs. You can pass these variables to the application as URL parameters.
**Example: "resultType":**
[Your path]de-2025/app/election/bars.html***?resultType=results***

Please note that only the first parameter is appended to the URL with a "?". All subsequent parameters must be separated by an "**&**":
[Your path]de-2025/app/election/bars.html**?*resultType=results*&*displayParties=6***

As a hosting customer, you'll find most of these options available for configuration on your setup page.
If parameter names have changed compared to the previous ElectionsLive widgets, we'll note this with "→ formerly [old name]".

You can pass the following parameters via the URL

| **Name** | **Options** (green= Default) | **Widgets** | **Description** |
|---|---|---|---|
| *allowedSourceIds* → formerly *singleSource* | "" "source2, source6" | All, except for surveys-only widgets and map | You can use *allowedSourceIds* to specify the IDs of the sources whose results should be displayed in the widgets. For example, if you want to display only ARD's forecasts and projections for a state election, pass *&allowedSourceIds=f912c49d-0a82-49b1-8742-54ae7c3f2897* |

| | | | |
|---|---|---|---|
| | | | Please contact us to obtain the corresponding IDs.<br>The source that provides the **preliminary and final results will always be displayed** regardless.<br>**Note**: This parameter is **very restrictive**! If neither a result from the source nor a final result is available, nothing will be displayed.<br>If you only pass poll sources to the widgets before the election, no live sources will be displayed on election night until the (preliminary) final result is available. |
| *areaId*<br>→ formerly *area* | “”<br>“76”<br>... | Constituencies, strongholds, individual constituency | This parameter controls which constituency the widget should start with when displaying constituency results. The official ID of the corresponding constituency, as published by the federal/state election commissioners, is required. **For the “Single Constituency” widget, the parameter must be provided** so that the widget can display data. For “Constituencies” and “Strongholds,” the parameter is optional. In these cases, it causes the result to be displayed immediately. |
| *autoResetTime* | "15" | All, except for surveys-only widgets and map | Specifies the time period in minutes during which the application switches from "auto" to "results"; the default value is "15". For example: In the 15 minutes before the exit poll is expected (i.e., from 5:45 PM onwards), the application no longer displays the last survey, but the message "No data available yet". |
| *baseOnly* | “false”<br>“true” | Coalition calculator flat | This parameter allows you to create a very simplified view of the coalition calculator. The buttons are hidden and the bars are no longer clickable. This allows you to create multiple widgets with example coalitions and arrange them one below the other. |

| | | | |
|---|---|---|---|
| | | | This parameter must be supplemented by at least one other:<br>• ***coalitionParties*** (so that a coalition is displayed)<br>• *coalitionLabel* (optional, e.g. “Jamaica”)<br>• *hideTitle* (optional, if you want to display multiple coalition calculators but only one title)<br>• *hideHeader* (only. *hideTitle*)<br>• *hideFooter* (only. *hideTitle*)<br>• *showFooterSource* (only. *hideTitle*) |
| *coalitionLabel* | “” | Coalition calculator flat with *baseOnly=true* | This only works with the simplified version of the coalition calculator. You can use this parameter to give the selected coalition a custom name. For example: “Grand Coalition” instead of “CDU/CSU - SPD”. |
| *coalitionParties* | “”<br>“par_de_1, par_de_3” | Coalition calculator | By default, the coalition calculator starts without any parties preselected. If you want the widget to start with a coalition, you can specify the IDs of the desired parties in *coalitionParties*, separated by commas. If any of the parties has no seats, it will be ignored. |
| *colorMode* | “auto”<br>“dark”<br>“light” | All | Whether the widgets are displayed in dark or light mode depends on the user’s preferences. If you embed the widgets on pages that are set to a specific color mode, you can pass *colorMode* as “dark” or “light,” and the widgets will always be displayed accordingly, regardless of the user’s preference. |
| *customConfigUrl* | String | All | Absolute path to your config.json file, which you can use to configure additional settings (see Chapter 5). If you are a Push customer and have the directory structure described in Chapter 2 on your publication server, the application automatically searches for the config.json file |

| | | | |
|---|---|---|---|
| | | | using relative paths in custom (please ensure that appLocation=push is passed). You only need to specify the path if the file is not located in de_2025/custom/config. |
| *customStyleUrl* | | All | Absolute path to your style.css. If you are a Push customer and have the directory structure described in Chapter 2 on your publication server, the application will automatically search for style.css using relative paths in the custom directory (please ensure that appLocation=push is passed). You only need to specify the path if the file is not located in de_2025/custom/style. |
| *customClass* | String | All | Here you can specify a custom class name that the application is then accessible via. It will be added as a data-attribute to the HTML-tag. This allows you, for example, to apply different layouts to the same widget using your style.css file. You can use it like this: `[data-custom-class="your-class"]{}` |
| *dataSource*<br>→ formerly via *seasonArchive* | "live"<br>"sim" | All | Using *dataSource*, you can control whether **live results/polls** or **test data** should be loaded. During our simulation sessions, you will be asked to pass *&dataSource=sim* to the widget so that you can follow the simulation. Please make sure that *dataSource* is set to "live" when publishing, otherwise only test data will be displayed. |
| *decimalPlaces* | "1"<br>"2"<br>"3" | All widgets with percentage values | This parameter allows you to control the number of decimal places. By default, one decimal place is displayed, unless a party is just below the percentage threshold, in which case two decimal places are automatically displayed.<br>The widgets are set to a maximum of 3. |
| *displayParties* | "auto"<br>"all" | All, except for surveys-only | The *displayParties* setting allows you to specify the maximum number of parties to be displayed. |

| | | | |
|---|---|---|---|
| → formerly *maxParties* | Number | widgets and coalition calculators | The “auto” setting ensures that as many parties as possible are displayed, depending on the available space. For the “**Columns**” widget (vertical bars), this means that parties that do not fit are grouped under “Others.” If you do not want this to happen, please set *displayParties* to “**all**” so that **all parties are always displayed**. If there is not enough space available, users will need to scroll horizontally.<br>For the other widgets, all parties are always displayed unless you specify a fixed number. In that case, all remaining parties are added to the “Others”. If *displayParties* is set to 6, 5 parties and the “Others” will be displayed. |
| *footerLinkUrl* | Absolute path | All | You can pass a URL to a linked page to the widget. A clickable arrow will then appear below the widget, which the user can click to go to the corresponding page. Please also pass the following: *footerLinkText*. |
| *footerLinkText* | String | All | Use this to specify the link text that should appear below the image if you have provided the *footerLinkUrl*. |
| *footerLinkTarget* | “_blank”<br>“_parent”<br>“_top”<br>“_self” | All | If you have passed the *footerLinkUrl* parameter, you can use it to control which window the URL should open in. |
| *hideHeaderLogo* | “false”<br>“true” | All | If you do not want to display the logoof the election in the header, pass *hideHeaderLogo=true* to the application. |
| *hideHeader*<br>→ formerly *widgetHeader* with default value true | “false”<br>“true” | All | If you set *hideHeader=true*, the header with the election title and logo will be hidden. Please note that the Refresh button will also disappear! |

| | | | |
|---|---|---|---|
| *hideFooter* →formerly *widgetFooter* with default value true | "false"<br>"true" | All | If you set *hideFooter=true*, the footer will be hidden. If the widget displays candidate photos, the footer will remain visible because it contains the button for the image sources. Only the dpa logo will disappear. |
| *hidePastBar* | "false"<br>"true" | Bars, columns, maps | This parameter allows you to hide the comparison bar that displays the value of the previous election. |
| *hideTitle* | "false"<br>"true" | Coalition calculator flat | This parameter hides the header row containing the result type, source, and status in the flat version of the coalition calculator. You can use this in combination with *baseOnly* if, for example, you want to place multiple simplified coalition calculators one below the other and only one of them should display the header. |
| *infoButtonMode* | "default"<br>"open"<br>"hidden"<br>"text" | All | You can use the infoButtonMode parameter to control the behavior of the info button below the graph.<br>• ***default:*** The info popup is closed and will open when clicked.<br>• ***open:*** The popup is already open at startup, but can also be closed again using the button.<br>• ***hidden:*** The button is not displayed, and the information about the graphic is not accessible!<br>• ***text:*** The information is displayed directly and cannot be hidden. |
| *language* | Iso-String<br>"de"<br>"en"<br>... | All | The *language* parameter specifies the language in which the widget should be displayed. The available languages vary depending on the selection. |
| *officeId* | ""<br>"bg-pres" | All | This parameter is required for elections in which multiple offices are up for election. In Spain, for |

| | | | |
|---|---|---|---|
| | “es-se”<br>… | | example, elections are held for the Congress of Deputies and the Senate—in this case, the widget must be provided with either `officeId=es` or `officeId=es-se`, depending on which results are to be displayed. Widgets that we set up for you via the setup page or the `custom` directory (for push customers) already include this parameter. |
| *partyOrder* | “”<br>“par_de_1, par_de_3” | beams, columns | By default, the order of parties/results in the widgets is determined by the size of each party’s vote share. With *partyOrder*, you can specify a custom order (e.g., based on the results of the previous election).<br>To do this, pass the IDs of the parties in partyOrder, separated by commas, in the desired order. Parties that are not listed are placed after them, sorted by vote count. |
| *resultType* | "auto",<br>"results",<br>"polls" | All except for surveys-only widgets | - “**auto”:** Default: show polls first, then automatically switch to live results as soon as they become available<br>- **„results“**: ONLY live results, empty if no results are available<br>- **„polls“**: survey results only |
| *round* | "1" | All | This parameter is relevant for elections held in multiple rounds, such as the French presidential election. You can use the *round* parameter to specify which round should be displayed. This should only be used if, for example, you want to embed the widget in an article about the first round of the election. Otherwise, we update the rounds automatically. |
| *seatListDirection* | "vertical"<br>"horizontal" | Seats - semicircle | This setting lets you control how the list below the pie chart is displayed. If you set *seatListDirection* to “horizontal,” the values will be displayed side by side. If there isn’t enough |

| | | | |
|---|---|---|---|
| | | | space, slider buttons will appear that allow you to scroll left and right. |
| *showFinalOnly* | "false"<br>"true" | All result widgets | If you set *showFinalOnly* to "true", all sources containing projections will disappear from the source dropdown, and only the (preliminary) final result will be displayed **once it is available**. |
| *showFooterSource* | "false"<br>"true" | All result widgets except constituencies | If you want the source to be displayed below the graph, as in the previous widgets, pass *&showFooterSource=true* to the widget. |
| *showOfficeName* | "false"<br>"true" | All | In elections where multiple offices are up for election and the widget uses the officeId parameter (see above), the showOfficeName parameter can be used to control whether the name of the office should be displayed next to the widget name. Example: Spain, Senate election: "Seat Distribution – Senate." We also include this parameter in our embed codes/widget URLs for the corresponding elections. |
| *showTrendResults* | "false"<br>"true" | US-Senate, US-House | The overall results for the Senate and House in the U.S. elections show only seats from states that have been definitively "called." If you also want to include results from states that have not yet been "called," set *showTrendResults* to *true*. |
| *sourceId* | UUID | All with source selection | If you want the widget to always start with a specific source, you can provide the corresponding UUID here. Please contact us to obtain the relevant ID.<br>If there is no result from the source, the parameter is ignored. |
| *updateInterval*<br>→ formerly *updateTime* | "60" | All | The default value for automatic updates is 60 seconds. You can adjust this value using a variable. There is no upper limit; the smallest possible value for the update cycle is 30. |

| | | | |
|---|---|---|---|
| | | | If you enter 0, no automatic updates will be performed and the update button will be hidden. |
| *valueType* | “percent” “seats” | “Flat beams” | The “Flat Bar” widget can display both percentages and seats. You can control this using the *valueType* parameter. |
| *voteType* | “first_vote” “second_vote” | Constituencies, strongholds, individual constituency results | If you pass *voteType=secondVote* to the widget, it will start directly by displaying the second votes. |

## 4.2 config.json

You can configure additional settings via the file **de-2025/custom/config/config.json**. We’ve already included some examples in the file config.example.json to help guide you.
As a hosting customer, you can download them from your setup page under “**Colors/Fonts**”.
You can then host these yourself (delete “.example” from the filename first) and pass them to the application via
**&customConfigUrl=*[absolute path to the folder].***

**Please note:** The file must be named “config.json”. And please specify only the path to the directory, without the filename.

You can also pass all the parameters listed in 4.1 as key/value pairs via the config.json file if you have general settings that should not change. If the same parameter is passed both via the config.json file and via the URL, the URL parameter takes precedence.
Example of parameters in the config.json:

```
{
    "hideHeader": true,
    "hideFooter": true
}
```

### 4.2.1 Changing Labels

If you want to change a label (e.g., a title), add an object named **labels** to your *config.json* file and enter the corresponding key-value pair:

```
{
    "labels": {
        "title_widget_bars": "Percentages"
    }
}
```

You can find the IDs of most labels by navigating to the corresponding element in your browser's developer console. There, you'll find the key in *data-label-id* and the original label in *data-label-value*. This is important if the label contains placeholders. An example of a label with a placeholder is "Absolute Majority from ###number###." Here, we use a script to replace "###number###" with the corresponding number. Therefore, please make sure to keep the placeholders unchanged when making adjustments.
If the data attributes for a label you want to change are not available, please contact us.

One exception is the **Title of the Election**, which we retrieve directly from our CMS. However, we also offer you the option to override this; to do so, enter the following key/value pair in the "labels" object:

```
"election_title": "Election in XY 2026"
```

### 4.2.2 Parties

You can customize the labels and party colors by adding a "parties" object and entering the party IDs you want to customize. You can find the ID for each party either through the developer console—for example, when you have the bar widget open—or by contacting us; we'd be happy to help.

We have defined a dark mode and a light mode color for each party. For the darkened or lightened party colors (e.g., in the comparison bars), we mix the colors with black or white depending on the color mode. However, you can also specify a color for each mode via the config.json file. If you only fill in individual color values, our default settings will be used for the rest.

If you only specify *color*, this color will only be used for light mode; for dark mode, the color we have defined will still be used. **We recommend specifying a color for each display mode**.
Please make sure that colors are specified as 6-digit values; short forms such as #fff or #000 do not work.

```
{
    "parties": {
        "par_de_5": {
            "color": "#BE3075",
```

```
            "color_dark": "#C52B68",
            "color_light": "#BE3075",
            "color_muted_dark": "",
            "color_muted_light": "",
            "name": "Die Linke",
            "abbreviation": "Linke"
        }
    }
}
```

If you want to display a large number of parties using a single color (e.g., all minor parties using the "Others" color), you can use the placeholder party "par_default" (in this example, instead of "par_de_5" and, of course, without "name" and "abbreviation").

**Note**: If "par_default" is found in config.json, all parties will be colored in the color specified there **unless they are explicitly listed individually**.

## 4.3 Configuring Your Own Layout

You can make all CSS customizations using the file de-2025/custom/style/style.css.
Hosting customers can download the file from their setup page.
**Your CSS customizations from the previous ElectionsLive widgets no longer work!**
Most settings are controlled via CSS variables, which we have already prepared for you in style.css (commented out). If you wish to make further customizations, you can find the corresponding classes using your browser's developer console.

The individual CSS variables are described in the comments. If anything is still unclear or if you have any further questions, please feel free to contact us.

**Please note:** Some variables (e.g., –dropdown__margin) are used in our CSS to calculate other values. In CSS, when performing addition or subtraction within the `calc()` function, you must not mix dimensionless numbers with units of length. Therefore, please make sure to always use the appropriate unit, even when setting a value to 0.
**Incorrect example:**
`--dropdown__margin: 0;` → All our calculations become invalid, and the CSS for the dropdown no longer works as it should.
**Correct example:**

`--dropdown__margin: 0px;` → Our calculations are not affected.

### 4.3.1 Changing the header image

If you want to remove the logo for the respective election, simply pass **&hideHeaderLogo=true** to the application (see above, chapter 4.1).

If you want to replace the logo with your own, follow these steps:
Hide our logo:

```
.header-logo-container img{
   display: none;
}
```

Insert the URL for your own logo as the background image:

```
.header-logo-container {
   background: url([ihre URL]/newLogo.png) no-repeat center;
}
```

If your logo is smaller or larger than ours, adjust the header padding to control the spacing from the title:

```
header {
   padding-left: 40px;
}
```

# 5. Test Data

In the run-up to the election, we provide test data that we use to simulate the various phases of election night.
We record each step of the simulation so that you can test every step across all widgets at any time.

As a **hosting customer**, you'll find this option on your setup page under "Settings" after clicking "Display widgets with dummy data." If you want to test the widgets after they've already been added to a test page, you can also follow the steps described in the next paragraph.
As a **Push customer**, the data is transferred to your S3 bucket. The test data is located in the *stage* directory, alongside our other feed directories, *live* and *sim*.
The "stage" directory contains subdirectories for each simulation step. So if there are 10 steps in the simulation, ranging from "Stage 0 – No Data" to "Stage 9 – Official Final Result," there will be 10 directories in *stage*, labeled *stage_0* through *stage_9*.

You can pass the respective steps to the widget using the ***dataSource*** parameter. This parameter controls the source directory from which the data is to be loaded.
For reference, here is a list of the possible settings for *dataSource*:

- dataSource=**live** → This is the default setting for the widgets and therefore does not need to be set separately. This, and only this, ensures that the latest results will be displayed on election night.
- dataSource=**sim** → You will be asked to set this in order to follow our simulations. This is where each simulation step is saved and then overwritten by the next one, so that the widgets automatically move on to the next step.
- dataSource=**stage/stage_3** → With this setting, you will receive test data for the initial projections by ARD and ZDF in most state elections. You can find out exactly what to expect in each stage by consulting the respective election simulation plan. You can then access the other steps using ***stage/stage_[number]***.

Chapter 4.1 explains the exact procedure for passing parameters to the widget.

**Please note:**

- In *stage_0* (no results), the widgets will in most cases **not be empty** but will display polls (if available). This is because the switch to "No results are available yet" is based on the time when the first live result is expected—for example, 6:00 p.m. on election night. We also generate this state in our simulations. Outside of these times, the widget displays the data it has—that is, results data if available, or polls otherwise.

- During a simulation session scheduled by us, the stage directories are rewritten. This means that at Stage 2 of the simulation, the *stage_9* directory (“Official Final Results”) does not yet exist or is empty. This ensures that any problems observed during the simulation can be easily traced. Once the simulation is complete, all directories are repopulated.
- **Important:** When publishing the widgets, please make sure that *dataSource* has either been completely removed from the URL or is set to *live*. This is the only way to publish live results.

# 6. Logging

Since there are now numerous different logging tools in use, we no longer offer direct integration with Google Analytics.
Instead, when the widgets launch and with every click, we send an object via postMessage that you can process individually. Unlike the object we send for height adjustment (see Section 3.3), this object lacks the height attribute; instead, it contains loggingData.
Here's how you can receive the object:

```
<script type='text/javascript'>
window.addEventListener('message',function(_evt) {
        if (_evt.data.sentinel === 'dpa' && _evt.data.type === 'logging') {
            // Process/forward object
        }
})
</script>
```

When the **widget starts**, you receive an object containing basic information and the interactionType "**view**"; every time a button within the **widget is clicked**, you receive an object with the interactionType "**click**" and information about the button that was clicked.

Example of the **coalition calculator at startup**:

```
{
    "sentinel": "dpa",
    "type": "logging", // Type of PostMessage
    "embed": "dpa_app_coa", // ID of the iframe, if passed to the widget
    "loggingData": {
        "referrer": "[URL of the embedding page]",
        "location": "[Host]/election/coalition_calculator.html?eventId=de_2025",
        "product": "dpa-electionslive",
        "eventId": "de_2025", // ID of the election
        "eventName": "", // "2025 Federal Election" → always empty under "view"
        "widget": "coalition_calculator", // ID of the widget
        "device": "desktop", // or "mobile", if touch is possible
        "btnTarget": "", // Button action, blank for "view"
        "btnFunction": "", // Button function, blank for "view"
        "interactionType": "view" //
    }
}
```

Example of the **coalition calculator when clicking on "CDU"**:

```
{
    "sentinel": "dpa",
    "type": "logging",
    "embed": "dpa_app_coa",
    "loggingData": {
        "referrer": "[URL of the embedding page]",
        "location": "[Host]/election/coalition_calculator.html?eventId=de_2025",
        "product": "dpa-electionslive",
        "eventId": "de_2025",
        "eventName": "Bundestagswahl 2025",
        "widget": "coalition_calculator",
        "device": "desktop",
        "btnTarget": "add",
        "btnFunction": "add/remove CDU to/from coalition",
        "interactionType": "click"
    }
}
```

# 7. Accessibility

Our widgets are designed to be accessible to as many users as possible, regardless of any disabilities they may have.
Key factors in this regard include colors, contrast, and font size. Therefore, when customizing these elements, please ensure that the widgets remain accessible to everyone.

# 8. GDPR

None of our widgets collect personal data on our end. **All of our applications are GDPR-compliant**. If you integrate Google Analytics, IVW tracking, or other processes that collect personal data into our widgets, please provide users with the standard notices and opt-out options.