# EUWA Wrapper Interface

The **EUWA Wrapper Interface** (also called *wrapper* below) provides a single point of entry and communication for the End User Web Applications (EUWA). It allows the developers to initiate the EUWA fast and to communicate with them, without managing the loading of the applications e.g. if you want to start chat with a custom button or add configuration/variables (runtime) from your web site.

The package is published in our public NPM Registry as@puzzel/euwa-wrapper

# Interface

```
declare interface Config {
    customerKey: string,
    configId: string
}

declare interface Options {
    settings: ApplicationSettings,
    hooks: Hooks
}

declare interface ApplicationList {
    [app: string]: string
}

declare interface Hooks {
    [hook: string]: Function
}

declare interface ApplicationBridge {
    api: ApplicationAPI,
    publish: (event: string, ...data: any) => void,
    subscribe: (event: string, callback: Function) => void,
}

declare interface ApplicationSettings {
    [app: string]: object
}

declare interface ApplicationAPI {
    [method: string]: Function
}

declare class EUWA {
    static APPS: ApplicationList

    constructor({customerKey, configId}: Config, {settings, hooks}: Options);
    getApplication(id: string): Promise<ApplicationBridge>;
    getApplicationBeforeLoad(id: string): ApplicationBridge;
}
```

# Connecting to the NPM Registry

A .npmrc file should be created either in your project or on user level. Read more about .npmrc on https://docs.npmjs.com/cli/v6/configuring-npm/npmrc

The following lines should be added:

```
@puzzel:registry=https://puzzel.pkgs.visualstudio.com/public/_packaging/main/npm/registry/
always-auth=true
```

# Basic usage

The basic usage will load the EUWA, with configuration set from Puzzel's Administration Portal.

```
import { EUWA } from '@puzzel/euwa-wrapper';

new EUWA({ configId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx', customerKey: 123456 });
```

## API Usage

The instantiation of the EUWA class will return a simple API, allowing to obtain access to any End-User Web Application context.

```
import {EUWA} from '@puzzel/euwa-wrapper';

const euwa = new EUWA({
    configId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx',
    customerKey: 123456
});

// Subscribe to click event on your start button
document.querySelector('#your-start-button').addEventListener('click', async () => {
    // Wait for the EUWA loader to run
    await euwa.load();

    // Get the chat application context
    const chat = euwa.getApplication(EUWA.APPS.CHAT);

    // Use the Chat's API to retrieve it's state
    const state = chat.api.getState();

    // Start a chat, if the user is not already in session
    if (!state.isConnected) {
        chat.api.startChat();
    }
});
```

## Adding variables

You can add and change system and custom variables from the web site to the chat session.

```
import {EUWA} from '@puzzel/euwa-wrapper';

const euwa = new EUWA({
    configId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx',
    customerKey: 123456
});

(async () => {
    // Wait for the EUWA loader to run
    await euwa.load();

    // Get the chat application context
    const chat = euwa.getApplication(EUWA.APPS.CHAT);

    // Update some system variables
    chat.api.updateSystemVariables({
        enteredFormName: 'James Bond',
        enteredChatId: 'bond@mail.mi6.co.uk',
        enteredFormIssue: 'I need a new mission, please.',
        selectedQueueKey: 'q_cookies_problems',
        timeId2Map: 'cookiesQueueWorkingTime'
    });

    // Update or set some custom variables
    chat.api.updateVariables({
        NewVariable: 'Some Value'
    });
})();
```

# Using hooks

onBeforeLoad
This allows you to subscribe to events or do other actions, before the applications are loaded. However, since the different application's API-s are defined by the application itself, they will not be available - only publish/subscribe interface is available.

> Note
>
> The EUWA's getApplicationBeforeLoad method is specifically designed to be used with specifically this hook. It will not wait for application's load and return it's basic communication interface - events.

```
import {EUWA} from '@puzzel/euwa-wrapper';

const euwa = new EUWA({
    configId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx',
    customerKey: 123456
}, {
    hooks: {
        // All hooks accept functions
        onBeforeLoad: subscribeToChatInit
    }
});

function subscribeToChatInit() {
    // Get the Chat's event interface
    const chat = euwa.getApplicationBeforeLoad(EUWA.APPS.CHAT);

    // Subscribe to chatInit* event
    chat.subscribe('chatInit', data => {
        console.log('Chat Init Data:', data);
    });
}
```

** A complete list of events can be found in our Chat Front End API article.*

# Overriding Settings

```
import {EUWA} from '@puzzel/euwa-wrapper';

const euwa = new EUWA({
    configId: 'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx',
    customerKey: 123456
}, {
    settings: {
        // Application name as first-level property
        [EUWA.APPS.CHAT]: {
            // Properties names as listed in Chat Admin
            showForm: false
        }
    }
});
```

 