Contents

1. Genesis toolbox

1.1 Automatic updates and versioning

1.1.1 Requirements:

- Trough GitLab.
- Release version
- Debug version
- CI/DI

1.2 User access on specific application functionalities, for instance buttons.

1.2.1 Requirements:

- Created in a separate .NetFramework class library.
- User´s functions are loaded from a web service.
- User´s functions are hold while application is running, or user is in session.
- Checking function against user´s functions.
- Clearing current user functions.
- !TODO: defining a way how should be this functionality bound to the login form of the application.

1.2.2 Implementation

- `class SoftwareFunctions.cs`
  - `Result Load(short appId, string username, string password, int timeout = default)` – available functions can be stored internal as some kind collection or as Enum. Timeout parameter is a time in milliseconds or standard default value after the software functions collection is cleared automatically (timer starts also automatically). Username and Password have no need of validation since they are checked by the web service. Whit appId parameter are searched all functions available for the app and its release version.
  - `bool CanAccess(string function) –` where the function name is checked against the existing user functions.
  - `void Clear() –` removes all user functions.
- `class Result.cs` – is a class that returns a state to the caller that holds possible errors during the operation process and its success.
  - `bool Succeeded { get; set; }`

- `IDictionary<string, string> Errors { get; set; }`