ifallfalse – Compare string against set of strings

Dennis Chen
proofprogram@gmail.com

v1.0.0, v. 2021/07/01*

Abstract

The ifallfalse package is a package that allows you to check whether a string is contained within another set of strings, and perform an action if it is not.

1 Usage

The package provides an ifallfalse environment and a macro `\orcheck` to be used inside the ifallfalse environment.

`ifallfalse` To set up an ifallfalse environment, simply write

```
\begin{ifallfalse}{string}{action}

\end{ifallfalse}
```

string will be compared to the set of strings, and if string does not match the set of strings, action will be executed.

`\orcheck`

To add strings to the set that string will be compared to, we must write `\orcheck{setstring}` inside the corresponding ifallfalse environment. Then, action will not execute if string matches setstring or any arguments of previous `\orcheck` declarations.

If no `\orcheck` declarations exist, then action will always be executed.

1.1 Error Checking

The package checks whether the macro `\orcheck` is used inside an ifallfalse environment. If it is not, the package throws an error.

*`https://github.com/chennisden/ifallfalse`

2 Example

Here is a simple example to demonstrate how ifallfalse is used.

```
one
```

3 Implementation

These are the implementation details of package ifallfalse. Because the package is so short, we can explain everything.

`ifallfalse` When setting up ifallfalse, we locally define the `\comparedstring` macro with the first argument that the environment takes in. This is what will be compared against all the strings passed in through the `\orcheck` declarations inside the environment.

Then, we define our body of logic (which we will be adding onto through `\orcheck`) to just initially consist of the action we would like to perform if `\comparedstring` matches none of the strings passed in through `\orcheck`.

```
\newenvironment{allfalse}[2]
{
    \def\comparedstring{#1}
    \def\logicbody{#2}
}
{
    \logicbody
}
```

`\orcheck` We first save allfalse to a macro so we can use `\ifx` to compare the current environment name against it. If we can, then we add some following (somewhat convoluted) code to `\logicbody`. I will explain what each piece of it does, though not in the order the pieces of code appear.

- `\ifx\@currenvir\@allfalsename` evaluates to true if the current environment (whose name is saved to the macro `\@currenvir`) matches the name of `\@allfalsename`, or allfalse.
- If it evaluates to false, the package throws an error.
- The line `\pdfstrcmp{\comparedstring}{#1}=0` evaluates to true when put with `\ifnum` if the two arguments passed into `\pdfstrcmp` are equal, because `pdfstrcmp` compares their lexographical order and returns 0 if the two strings are lexographically equivalent.
- Thus, we can treat `\ifnum\pdfstrcmp{\comparedstring}{#1}=0` as an expression that evaluates to true if `\comparedstring` and `#1` match, and false otherwise.

- When all is said and done, the logic reduces to something of the form

```
\if\else
\if\else
\ldots action
\fi\ldots \fi
```

  Logically, action will only execute if all the conditions are false; in other words, it will only execute if `\pdfstrcmp{#1}` does not match any of the strings passed in via `\orcheck`. This is because each `\else` branch must execute.

```
\newcommand*\@allfalsename{allfalse}

\newcommand{\orcheck}[1]{
    \ifx\@currenvir\@allfalsename
        \protected@edef\logicbody{\ifnum\pdfstrcmp{\comparedstring}{#1}=0\else\logicbody\fi}
    \else
        \PackageError{ifallfalse}{\protect\orcheck\space should be nested within the allfals
    \fi
}
```