The spdef Package

D. P. Story

Released June 28, 2012

```
⟨*package⟩
```

Description and Usage. This is a short package to create the `\ifsmartphone` switch. The package is designed to be introduced early in the file, even before `\documentclass`. I use `\RequirePackage`, like so.

```
   \RequirePackage[ph]{spdef}
   \documentclass[\ifsmartphone12pt\else10pt\fi]{article}
   \usepackage[fleqn]{amsmath}
   \usepackage[pdf,myconfigi,nopoints,answerkey]{eqexam}
\ifsmartphone
    \usepackage[smartphone,nomaketitle,useforms]{aeb_mobile}
\fi
```

When you use `\usepackage`, there is an error which says no class file has been used, but apparently it is file to have `\RequirePackage` before a class file; consequently, it can be used to adjust the point size of the document.

Version 1.2 of aeb_mobile works better with spdef. Now if `\ifsmartphone` is false, aeb_mobile does an early exit; consequently, surrounding it with the construct `\ifsmartphone`...`\fi` is no longer needed:

```
   \RequirePackage[ph]{spdef}
   \documentclass[\ifsmartphone12pt\else10pt\fi]{article}
   \usepackage[fleqn]{amsmath}
   \usepackage[pdf,myconfigi,nopoints,answerkey]{eqexam}
   \usepackage[smartphone,nomaketitle,useforms]{aeb_mobile}
```

See Section 2 for more details and additional options.

Another feature of this package is the automatic creation of Boolean switches. If you say

```
   \RequirePackage[use=myswitch]{spdef}
```

a new switch `\ifmyswitch` is created and given a value of true. If you say

```
   \RequirePackage[!use=myswitch]{spdef}
```

a new switch `\ifmyswitch` is created and given a value of false. See Section 3 for more details.

Of course, if you do not need to introduce spdef before the class in included, you can use the standard `\usepackage` command.

1 The Code

We begin by requiring `kvoptions`, this package does not test the the presence of a class file, so we can use it. It allows us to define key-values as options of the package.

```
\RequirePackage{kvoptions}[2009/07/21]
```

The package is primarily intended for use with the aeb_mobile package, for formatting document for the smartphone, but I've since developed other applications of a package that is introduced early, see the definition of the `use` key.

```
\newif\ifsmartphone  \smartphonefalse
```

2 smartphone options

We offer two options `ph` and `pa`, additional options for other devices may be defined.

- `ph` sets the `\ifsmartphone` switch to true. The name `ph` stands for phone.
- `pa` sets the `\ifsmartphone` switch to false. The name of the option, `pa`, stands for paper.

`ph` Option for phone: sets the switch `\ifsmartphone` to `true`.

`pa` Option for paper: sets the switch `\ifsmartphone` to `false`.

```
\DeclareVoidOption{ph}{\smartphonetrue}
\DeclareVoidOption{pa}{\smartphonefalse}
```

`!ph` `!pa` It's easy enough, lets do negatives of the two option above. `!ph` is the same as `pa` and `!pa` is the same as `ph`.

```
\DeclareVoidOption{!ph}{\smartphonefalse}
\DeclareVoidOption{!pa}{\smartphonetrue}
```

3 Defining Boolean switches on the fly

Based on my own work, I've added in two more options `use` and `!use`. Suppose we want to create a switch, say `\ifforinstr`, we can say,

```
\usepackage[use=forinstr]{spdef}
```

The spdef package would create a new Boolean `\ifforinstr` and assign it a value of `true`. If you want to compile the document with `\ifforinstr` having a value of `false`, we would modify the above options like so,

```
\usepackage[!use=forinstr]{spdef}
```

`use` The `use=<switch>` is a way to define/use a switch early in the compiling of the document, even before the document class is declared. The code below creates the switch `\if<switch>`, and sets it to `true`. The document that uses this switch should have this code in it:

```
\@ifundefined{if<switch>}{\newif\if<switch>\<switch>false}{}
```

We can set this switch to `true` through the spdef package, otherwise, its value is `false`.

`!use` Given my last remarks on the `use` key, as a convenience, we declare the option `!use`. It does the same as `use`; it creates the switch but sets it to `false`. That way, you can say `use=useendnotes` and `\ifuseendnotes` is `true`, or, by prefixing `use` with an `!`, like so, `!use=useendnotes`, spdef defines/sets `\ifuseendnotes` to `false`.

```
\define@key{spdef}{use}{\@ifundefined{#1}{%
    \expandafter\newif\csname if#1\endcsname}{}\csname#1true\endcsname}
\define@key{spdef}{!use}{\@ifundefined{#1}{%
    \expandafter\newif\csname if#1\endcsname}{}\csname#1false\endcsname}
```

If the key is not used, back in the document that uses the switch,

```
\@ifundefined{if<switch>}{\newif\if<switch>\<switch>false}{}
```

will set this value to `false`; in this case, you need to explicitly set the value of the switch yourself.

```
\ProcessKeyvalOptions{spdef}
```

`\ifsp` `\ifsp`⟨*TRUE*⟩⟨*FALSE*⟩ is a convenience command for the `\ifsmarphone` switch. It takes two arguments, the first one if the `\ifsmartphone` is true, the second one if not.

```
\def\ifsp@default#1#2{\ifsmartphone
    \expandafter\def\csname sp@next\endcsname{#1}\else
    \expandafter\def\csname sp@next\endcsname{#2}\fi\sp@next}
\def\ifsp@expand#1#2{\ifsmartphone#1\else#2\fi}
\let\ifsp\ifsp@default
```

```
⟨/package⟩
```