The beamer class: Controlling overlays

Joseph Wright

There was a question recently on the TEX StackExchange site (Gil, 2014) about the details of how slide overlays work in the beamer class (Tantau, Wright, and Miletić, 2013). The question itself was about a particular input syntax, but it prompted me to think that a slightly more general examination of how overlays would be helpful to beamer users.

A word of warning before I start: don't overdo overlays! Having text or graphics appear or disappear on a slide can be useful but is easy to over-use. I'm going to focus on the mechanics here, but that doesn't mean that they should be used in every beamer frame you create.

1 Overlay basics

Before we get into the detail of how beamer deals with overlays, I'll first give a bit of background to what they are. The beamer class is built around the idea of frames:

```
\begin{frame}
  \frametitle{A title}
  % Frame content
\end{frame}
```

which can produce one or more slides: individual pages of output that will appear on the screen. These separate slides within a frame are created using overlays, which is the way the beamer manual describes the idea of having the content of individual slides varying. Overlays are "contained" within a single frame: when we start a new `frame`, any overlays from the previous one stop applying.

The most basic way to create overlays is to explicitly set up individual items to appear on a particular slide within the frame. That's done using the (optional) overlay argument that beamer enables for many document components; this overlay specification is given in angle brackets. The classic example is a list, where the items can be made to appear one at a time.

```
\begin{frame}
  \begin{itemize}
    \item<1-> Visible from the 1st slide
    \item<2-> Visible from the 2nd slide
    \item<3-> Visible from the 3rd slide
    ...
  \end{itemize}
\end{frame}
```

As you can see, the overlay specification here is simply the first slide number we want the item to be on followed by a `-` to indicate "and following slides". We can make things more specific by giving only a single slide number, giving an ending slide number and so on.

```
\begin{frame}
  \begin{itemize}
    \item<1> Visible on the 1st only
    \item<-3> Visible on the
      1st to 3rd slides
    \item<2-4,6> Visible on the
      2nd to 4th slides, and the 6th slide
  \end{itemize}
\end{frame}
```

The syntax is quite powerful, but there are at least a couple of issues. First, the slide numbers are hard-coded. That means that if I want to add something else in before the first item I've got to renumber everything. Secondly, I'm having to repeat myself. Luckily, beamer offers a way to address both of these concerns.

2 Auto-incrementing the overlay

The first tool beamer offers is the special symbol `+` in overlay specifications. This is used as a place holder for the "current overlay", and is automatically incremented by the class. To see it in action, I'll rewrite the first overlay example without any fixed numbers.

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
    \item<+-> Visible from the 3rd slide
    ...
  \end{itemize}
\end{frame}
```

What's happening here? Each time beamer finds an overlay specification, it automatically replaces all of the `+` symbols with the current overlay number. It then advances the overlay number by 1. So in the above example, the first `+` is replaced by a `1`, the second by a `2` and the third by a `3`. So we get the same behaviour as in the hard-coded case, but this time if I add another item at the start of the list I don't have to renumber everything.

There are of course a few things to notice. The first overlay in a frame is number 1, and that's what beamer sets the counter to at the start of each frame. To get the second item in the list to appear on slide 2, we still require an overlay specification for the first item: I could have skipped the `<1->` in the hard-coded example and nothing would have changed. The second point is that *every* `+` in an overlay specification gets replaced by a given value. We'll see later there

are places you might accidentally add a + to mean "advance by 1": don't do that!

3 Reducing redundancy

Using the + approach has made our overlays flexible, but I've still had to be repetitive. Handily, beamer helps out there too by adding an optional argument to the list which inserts an overlay specification for each line:

```
\begin{frame}
  \begin{itemize}[<+->]
    \item Visible from the 1st slide
    \item Visible from the 2nd slide
    \item Visible from the 3rd slide
    ...
  \end{itemize}
\end{frame}
```

Notice that this is needs to be inside the "normal" [...] set up for an optional argument. Applying an overlay to every item might not be exactly what you want: you can still override individual lines in the standard way.

```
\begin{frame}
  \begin{itemize}[<+->]
    \item Visible from the 1st slide
    \item Visible from the 2nd slide
    \item Visible from the 3rd slide
    \item<1-> Visible from the 1st slide
     ...
  \end{itemize}
\end{frame}
```

Remember not to overdo this effect: just because it's easy to reveal every list line by line doesn't mean you should!

4 Repeating the overlay number

The + syntax is powerful, but as it always increments the overlay number it doesn't allow us to remove the hard-coded numbers from a case such as

```
\begin{frame}
  \begin{itemize}
    \item<1-> Visible from the 1st slide
    \item<1-> Visible from the 1st slide
    \item<2-> Visible from the 2nd slide
    \item<2-> Visible from the 2nd slide
     ...
  \end{itemize}
\end{frame}
```

For this case, beamer offers another special symbol, a single period '.', as in:

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<.-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
    \item<.-> Visible from the 2nd slide
     ...
  \end{itemize}
\end{frame}
```

What happens here is that . can be read as "repeat the overlay number of the last +". So the two + overlay specifications create one slide each, while the two lines using . in the specification 'pick up' the overlay number of the preceding +. (The beamer manual describes the way this is actually done, but I suspect that's less clear than thinking of this as a repetition!)

Depending on the exact use case, you might want to combine this with the "reducing repeated code" optional argument, with <.-> as an override.

```
\begin{frame}
  \begin{itemize}[<+->]
    \item Visible from the 1st slide
    \item<.-> Visible from the 1st slide
    \item Visible from the 2nd slide
    \item<.-> Visible from the 2nd slide
     ...
  \end{itemize}
\end{frame}
```

5 Offsets

A combination of + and . can be used to convert many "hard-coded" overlay set ups into "relative" ones, where the slide numbers are generated by beamer without you having to work them out in advance. However, there are still cases it does not cover. To allow even more flexibility, beamer has the concept of an "offset": an adjustment to the number that is automatically inserted. Offset values are given in parentheses after the + or . symbol they apply to, for example:

```
\begin{frame}
  \begin{itemize}
    \item<+(1)-> Visible from the 2nd slide
    \item<+(1)-> Visible from the 3rd slide
    \item<+->    Visible from the 3rd slide
  \end{itemize}
\end{frame}
```

Notice that this adjustment only applies to the substitution, so both the second and third lines above end up as <3-> after the automatic replacement. If you try the demo, you'll also notice that none of the items appear on the first slide!

Perhaps a more realistic example for where an offset is useful is the case of revealing items "out of

order", where the full list makes sense in some other way. With hard-coded numbers this might read

```
\begin{frame}
  \begin{itemize}
    \item<1-> Visible from the 1st slide
    \item<2-> Visible from the 2nd slide
    \item<1-> Visible from the 1st slide
    \item<2-> Visible from the 2nd slide
    ...
  \end{itemize}
\end{frame}
```

which can be made "flexible" with a set up such as

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
    \item<.(-1)-> Visible from the
                  1st slide
    \item<.-> Visible from the 2nd slide
    ...
  \end{itemize}
\end{frame}
```

or the equivalent

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<.(1)-> Visible from the 2nd slide
    \item<.-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
    ...
  \end{itemize}
\end{frame}
```

As shown, we can use both positive and negative offsets, and these work equally well for `+` and `.` auto-generated values. You have to be slightly careful with negative offsets; while beamer will add additional slides for positive offsets, if you offset to below a final value of `0` then errors will crop up. With this rather advanced setup, which version is easiest for you to follow will be down to personal preference.

Notice that positive offsets do not include a `+` sign, but are just given as an unsigned integer: remember what I said earlier about all `+` symbols being replaced. If you try something like `<+(+1)>`, your presentation will compile but you'll have a lot of slides!

6 Pausing general text

The beamer class offers a very simple `\pause` command to split general material into overlays. A classic problem that people run into is combining that idea with the `+` approach to making overlays. For example, the following creates *four* slides:

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
  \end{itemize}
  \pause
  Text after the list
\end{frame}
```

If you read the beamer manual carefully, this is what is supposed to happen here, but the more important question is how to get what you (probably) want: three slides.

The answer is to use `\onslide`: the `\pause` command is by far the most basic way of making overlays, and simply doesn't "know" how to work with `+-`. In contrast, `\onslide` uses exactly the same syntax we've already seen for overlays:

```
\begin{frame}
  \begin{itemize}
    \item<+-> Visible from the 1st slide
    \item<+-> Visible from the 2nd slide
  \end{itemize}
  \onslide<+->
  Text after the list
\end{frame}
```

As we are then using the special `+` syntax for all of the overlays, everything is properly tied together and gives the expected result: three slides.

7 Summary

The beamer overlay feature can help you set up complex and flexible overlays to generate slides with dynamic content. By using the tools carefully, you can make your input easier to read and maintain.

References

Gil, Yossi. "Relative overlay specification in beamer?" `http://tex.stackexchange.com/q/154521`, 2014.

Tantau, Till, J. Wright, and V. Miletić. "The beamer class". Available from CTAN, `macros/latex/contrib/beamer`, 2013.

◇ Joseph Wright
2, Dowthorpe End
Earls Barton
Northampton
NN6 0NH
United Kingdom
`joseph.wright (at) morningstar2 dot co dot uk`