Android Spanned, SpannedString, Spannable, SpannableString and CharSequence

13

android android-textview

Android offers a variety of interfaces all related to text and strings: Spanned, SpannedString, Spannable, SpannableString and CharSequence.

I have used all of the above in various scenarios **usually** after using Html.fromHtml() to display linkable text inside a TextView in order to apply some styling to it.

I have tried to understand the purpose/usage of these interfaces from Android's official documentation and have failed as it is quite confusing.. What is the purpose of these interfaces? in which scenarios is it mostly common to use them? In which cases is it best to avoid using them? Are there any obvious performance impacts to be considered when using any one of them?

If anyone could provide a decent explanation it would be much appreciated.

Thank you

Tom R.
1,129●7●25

Asked
Jul 9 '13 at 11:22

Edited
Feb 2 at 8:07

1 Answer

order by votes

31

> What is the purpose of these interfaces?

CharSequence is a standard Java interface representing a sequence of characters. String is the most commonly-used concrete implementation of CharSequence, followed by StringBuilder.

Spanned is a CharSequence with "spans" indicating formatting to apply to portions of the text, where those spans cannot be modified.

Spannable is a Spanned, adding in the ability to modify the spans (to add or remove formatting), but *not* to modify the text itself.

SpannedString is a concrete implementation of the Spanned interface.

SpannableString is a concrete implementation of the Spannable interface.

> in which scenarios is it mostly common to use them?

When there is a method that returns one (e.g., getText() on an EditText) or when there is a method that takes one as a parameter (e.g., setText() on a TextView).

Your cited case of using Html.fromHtml() is perhaps the most common in conventional Android development, as a TextView with a Spanned is much lighter in weight than is a WebView. However, there are other use cases, such as:

- Highlighting search results
- Allowing users to enter in rich text, then using Html.toHtml() to persist that formatted text in an HTML rendition

> In which cases is it best to avoid using them?

They are singularly awful at combating baldness, snow removal, heat pump repair, making a soufflé, etc.

:-)

> Are there any obvious performance impacts to be considered when using any one of them?

Interfaces, by definition, do not have "performance impacts" -- they are merely a description of an API.

I am not aware that SpannableString is significantly slower than SpannedString at any particular operation. However, SpannableStringBuilder (which allows for manipulating the text in addition to the spans that format that text) may well be a bit slower than SpannableString or SpannedString for various things. Whether or not the performance differences are enough to matter will depend on usage, though.

share improve this answer

CommonsWare
400k 35 827 895

Answered
Jul 9 '13 at 11:41

You are awesome, thanks! – **Tom R.** Jul 9 '13 at 12:07

add a comment

Your Answer

log in

or

Name

Email

Home Page

By posting your answer, you agree to the privacy policy and terms of service.

Post Your Answer