

Advanced Text Layouts and Effects with Text Kit

Session 220

Aki Inoue

Digital Textmancer

These are confidential sessions—please refrain from streaming, blogging, or taking pictures

Advanced Text Layouts and Effects with Text Kit

Freedom of control over your text

Session 220

Aki Inoue

Digital Textmancer

These are confidential sessions—please refrain from streaming, blogging, or taking pictures

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda

- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Agenda



- Text effects
- Main Text Kit objects
- Text layout explained
- Customizing Text Layouts

Text Effects

Peter Hajas
UIKit Engineer



Happy Birthday, Nana!



nday, Nana!

Text Kit Classes

Text Kit Architecture



NSStringStorage

The quick brown fox *jumps over* the lazy *dog*

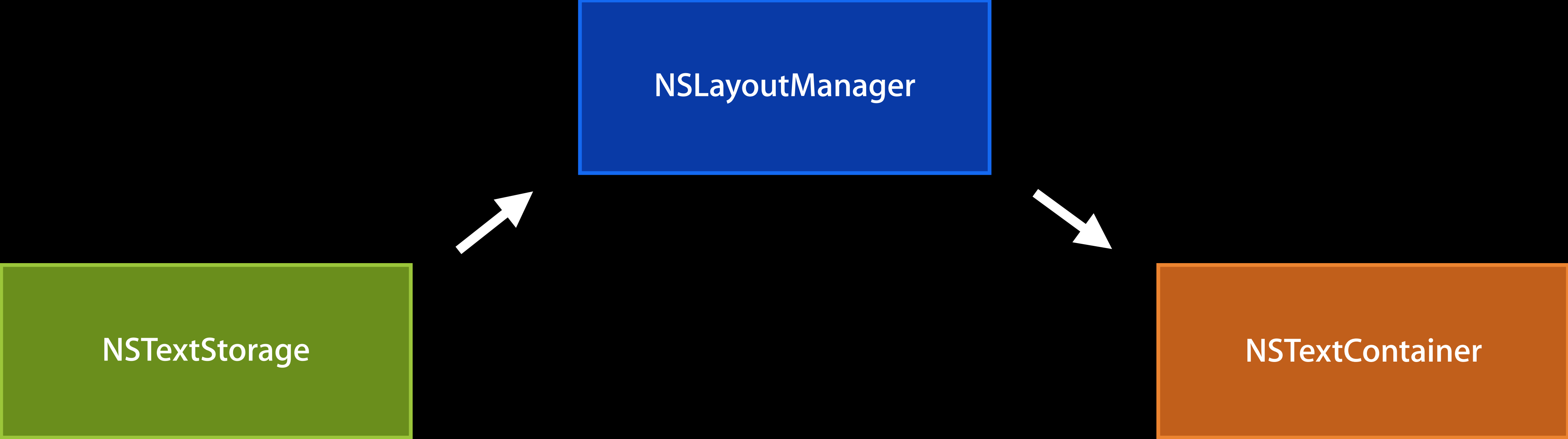
NSTextStorage



NSLayoutManager



NSTextContainer



NSTextStorage



NSLayoutManager



NSTextContainer

NSTextContainer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus lacinia pretium diam non tempor. Aenean mollis pellentesque lectus, vitae ultrices urna tincidunt eu. Mauris ullamcorper elementum pharetra. Donec imperdiet lacinia porttitor. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nulla lobortis tortor libero. Donec fringilla placerat lectus sed commodo. Nulla nisl nulla, feugiat eu sodales nec, semper non nibh. Nunc porta lacinia cursus. Vestibulum ultrices euismod euismod. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Quisque nec lectus id diam molestie consectetur sed sed ligula. Nulla non luctus nibh. Integer viverra posuere urna, vel volutpat eros eleifend in. Donec pharetra tincidunt lectus vitae luctus.

Ut semper vulputate quam in dictum. Maecenas lobortis porttitor lorem vel molestie. Nam eros orci, mattis ac placerat nec, blandit sed orci. In consequat convallis risus eu fermentum. Mauris accumsan lobortis porta. Nunc feugiat, leo et consequat varius, velit metus consectetur ante, in bibendum neque felis vel sapien. Fusce vel risus in tellus convallis facilisis. Nunc

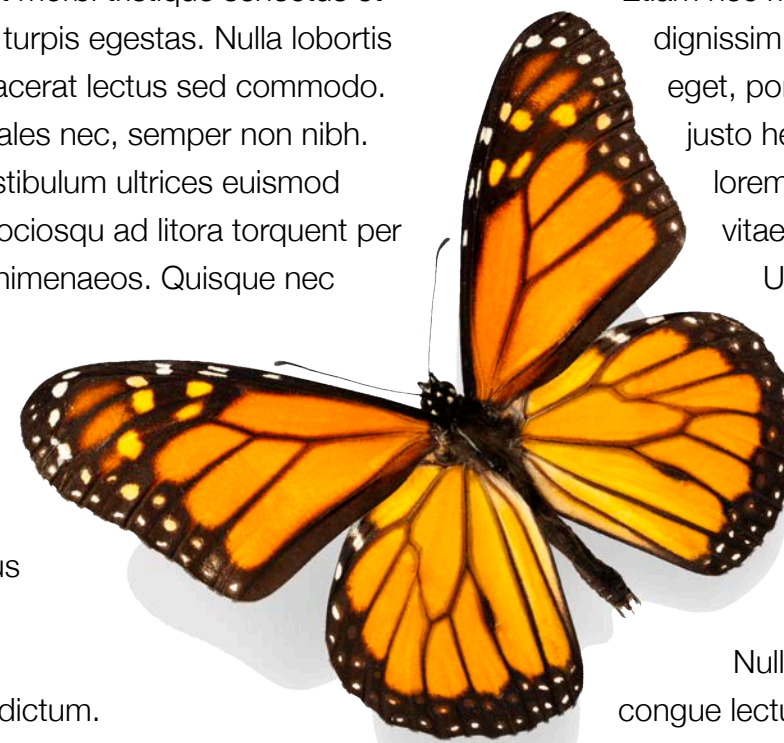
consectetur fringilla sem vel varius. Etiam cursus auctor tortor vitae dictum. Sed interdum fringilla orci, sed commodo magna ultricies fringilla. Donec eget convallis lacus.

Etiam nec mauris lacus. Cras mattis lobortis dignissim. Sed lorem turpis, feugiat at sodales eget, porta vel purus. Sed ullamcorper diam ac justo hendrerit porta. Aliquam sed erat ut lorem facilisis sollicitudin quis eget mi. Sed vitae massa id magna sagittis commodo.

Ut feugiat tincidunt purus, et imperdiet diam convallis vitae. Donec augue libero, blandit ut dapibus id, vulputate at velit. Morbi condimentum bibendum turpis, sed fermentum turpis ornare non.

In hac habitasse platea dictumst.

Nulla facilisi. Proin vel nibh mi, quis congue lectus. Etiam sit amet est nec quam iaculis lobortis sit amet eu leo. Nulla mollis feugiat quam, a interdum sapien pellentesque sed. Pellentesque eu sem ut elit fringilla scelerisque a vel leo. Aenean quis lacus eget massa condimentum adipiscing ac vitae sapien. Vivamus id nibh aliquet ante blandit varius ac lobortis nisl. Pellentesque turpis ante, consectetur egestas semper eget,



NSTextStorage



NSLayoutManager



NSTextContainer

Advanced Configuration



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue dui dolore te feugait nulla facilisi.

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum.

Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum formas humanitatis per seacula quarta decima et quinta decima. Eodem modo typi, qui nunc nobis videntur parum clari, fiant sollemnes in futurum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue dui dolore te feugait nulla facilisi.

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum.

Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum formas humanitatis per seacula quarta decima et quinta decima. Eodem modo typi, qui nunc nobis videntur parum clari, fiant sollemnes in futurum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Advanced Configuration



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exercitation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

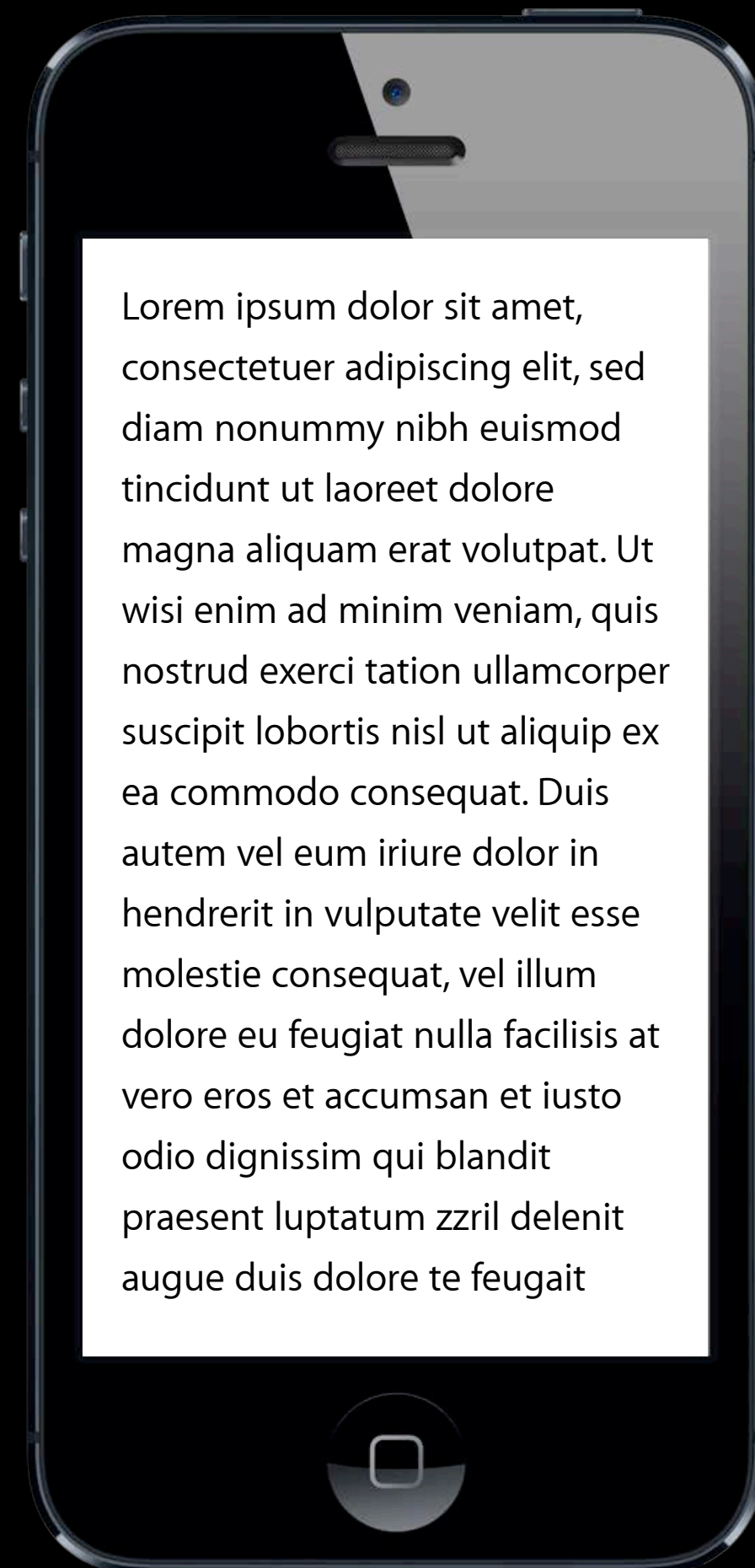
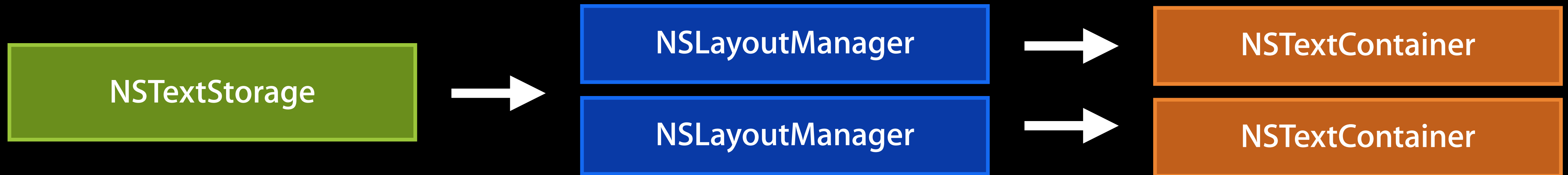
Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem.

Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum.

Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum formas humanitatis per seacula quarta decima et quinta decima. Eodem modo typi, qui nunc nobis videntur parum clari, fiant sollemnes in futurum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exercitation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Advanced Configuration



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue dui dolore te feugait nulla facilisi.

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum.

Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum

Demo

Multi page document

Text Layout Explained

NSLayoutManager

NSLayoutManager

- Controller of the text layout process

NSLayoutManager

- Controller of the text layout process
- Manages the layout information

NSLayoutManager

- Controller of the text layout process
- Manages the layout information
- All text layout information accessible

NSLayoutManager

- Controller of the text layout process
- Manages the layout information
- All text layout information accessible
- Flexible extensibility via subclassing and delegation

Text Layout

Text Layout =

**Text Layout =
Glyphs + Locations**

Glyphs

Glyphs

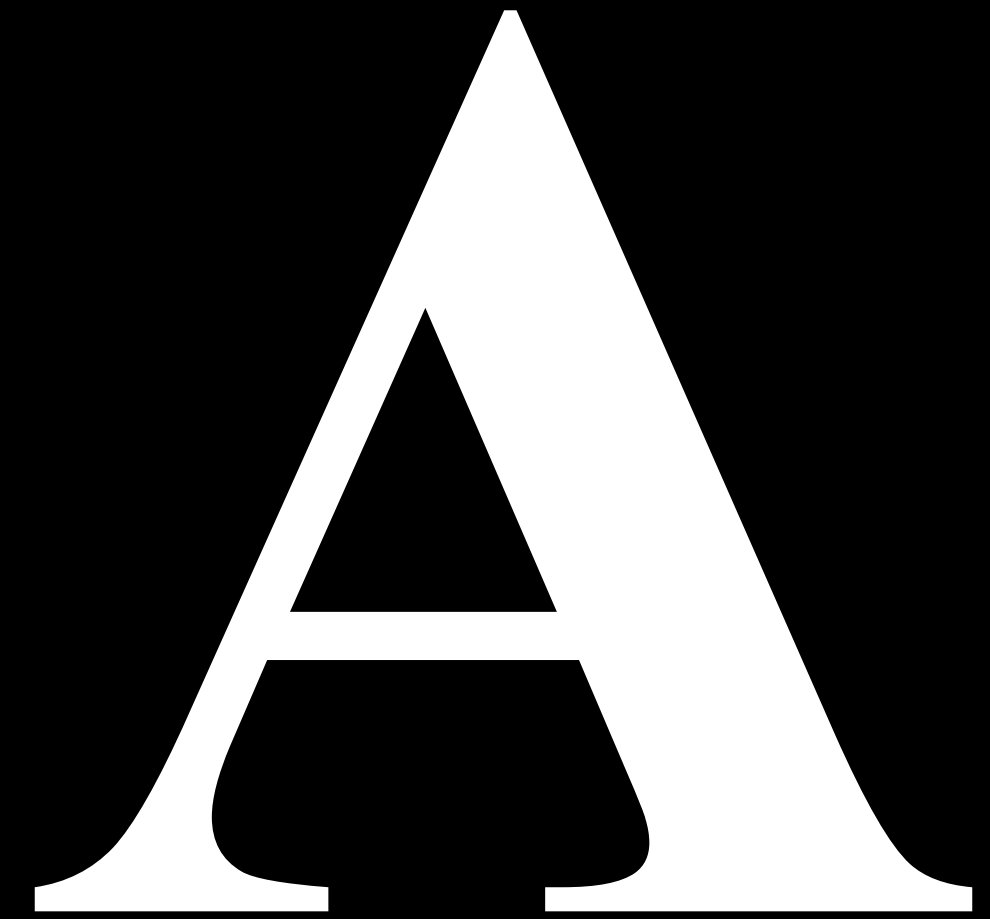
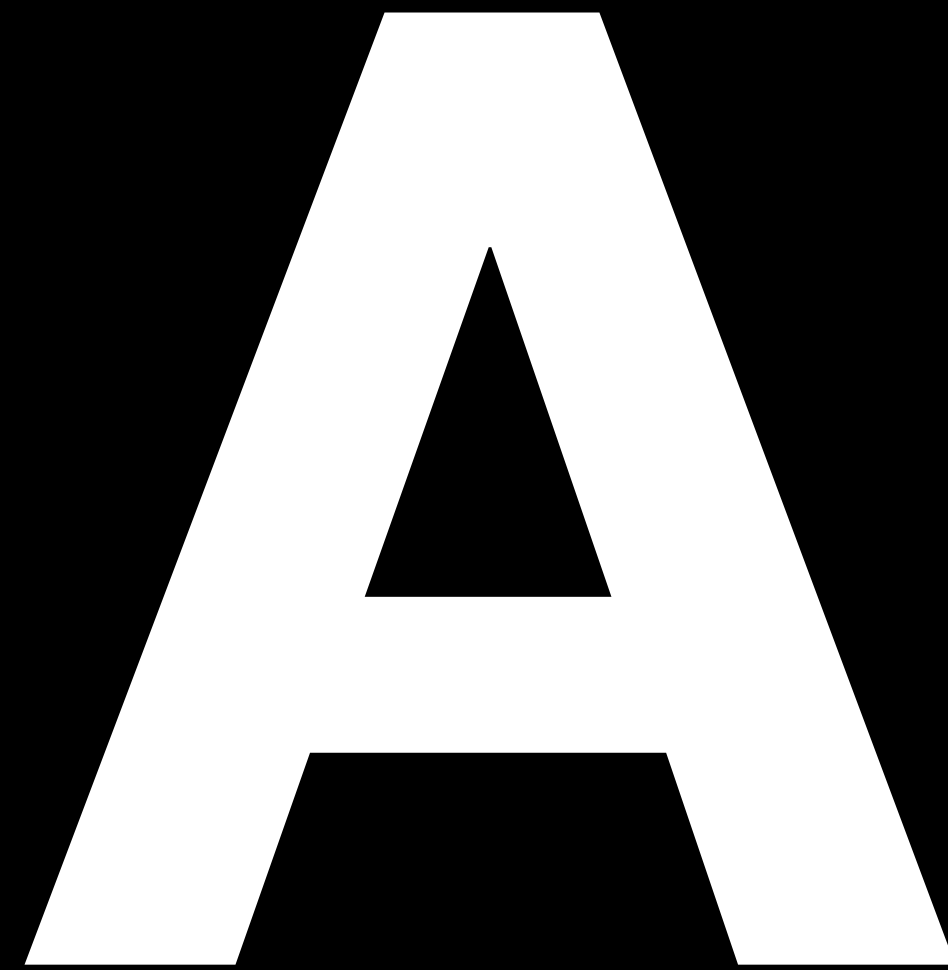
What is a Glyph?

What is a Glyph?

A Graphical Representation of Characters

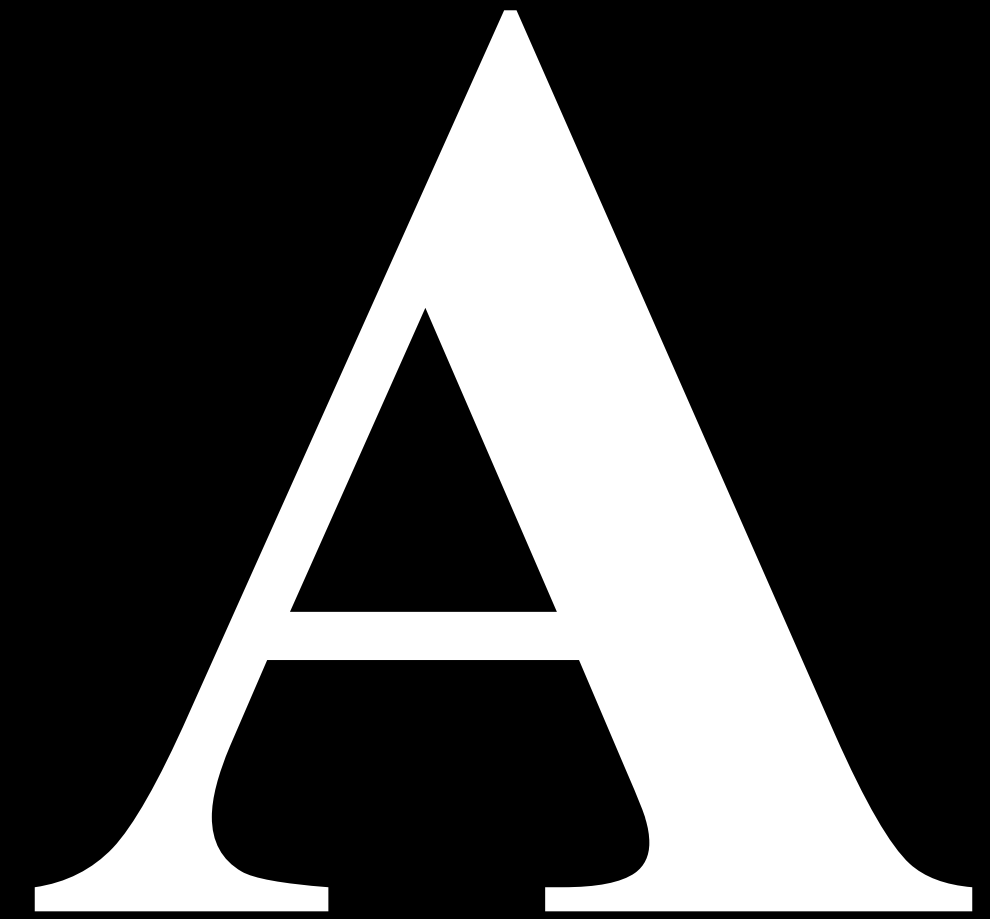
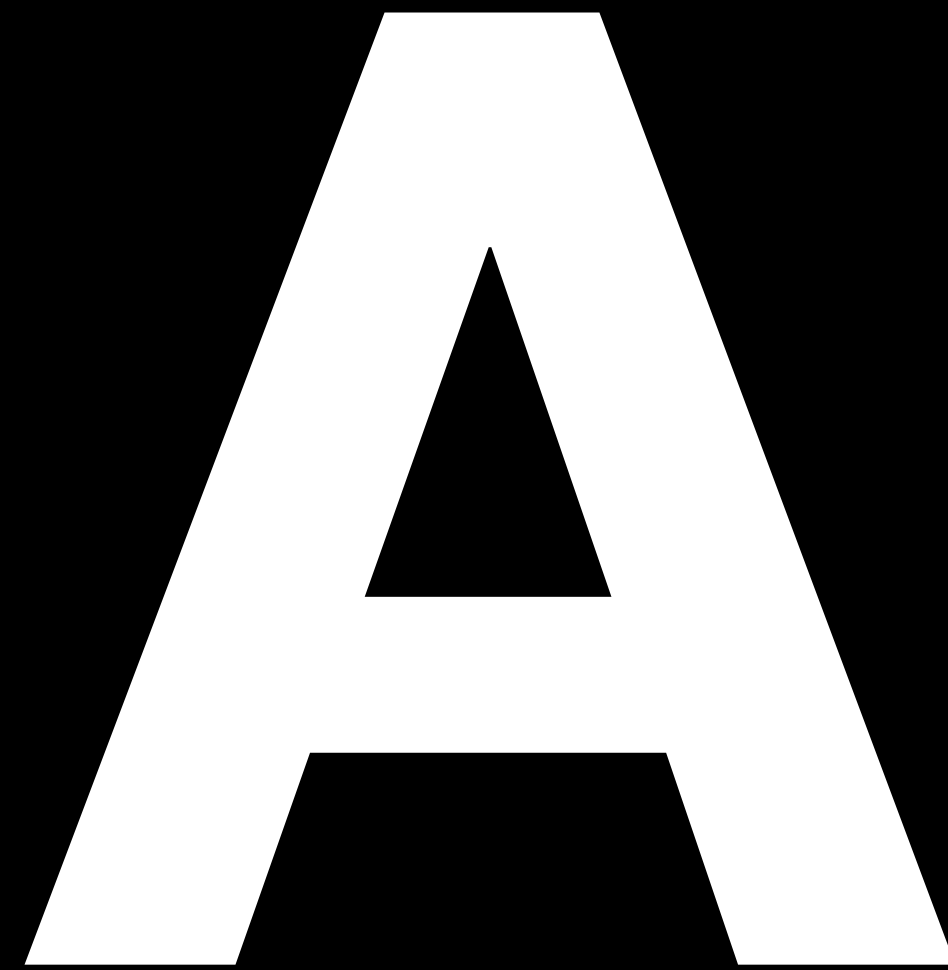
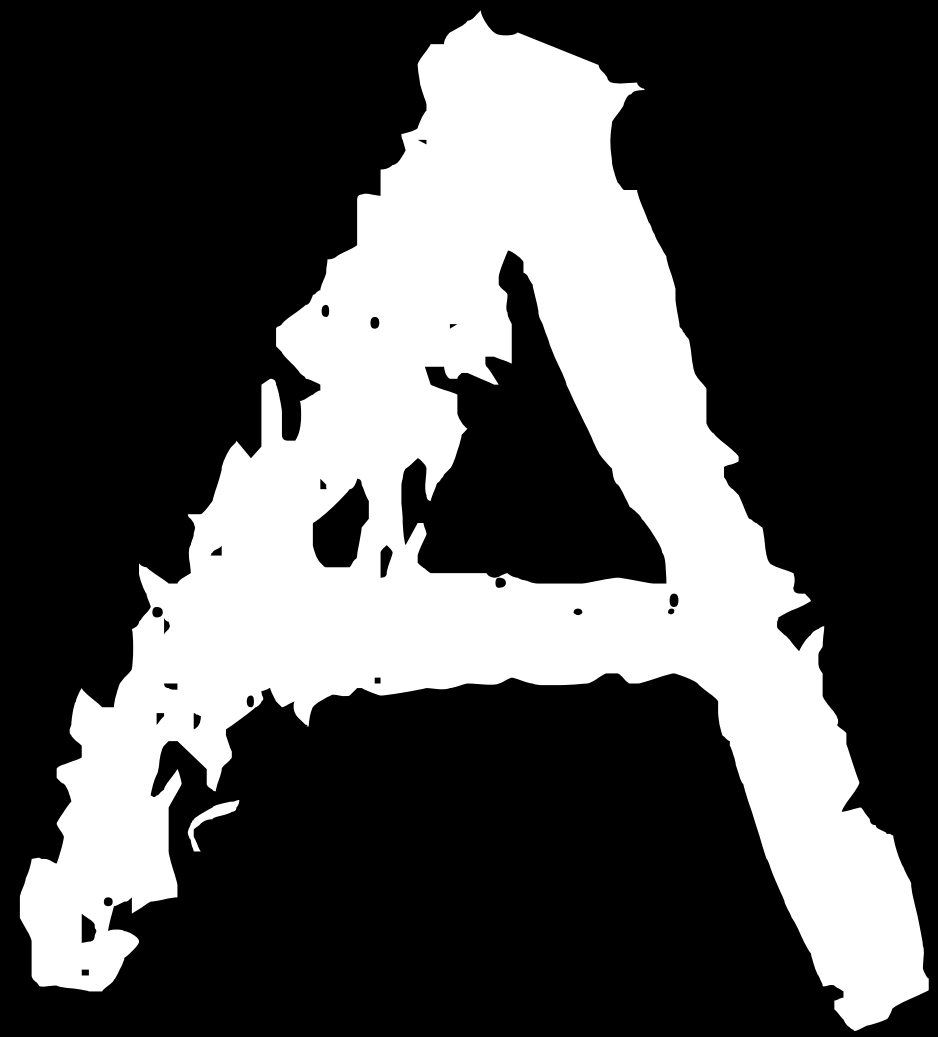
What is a Glyph?

A Graphical Representation of Characters



What is a Glyph?

- A graphical representation of characters
- Character + font -> glyph



What is a Glyph?

- A graphical representation of characters
- Character + font -> glyph
- Glyph IDs for the graphics systems: **CGGlyph**

Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering

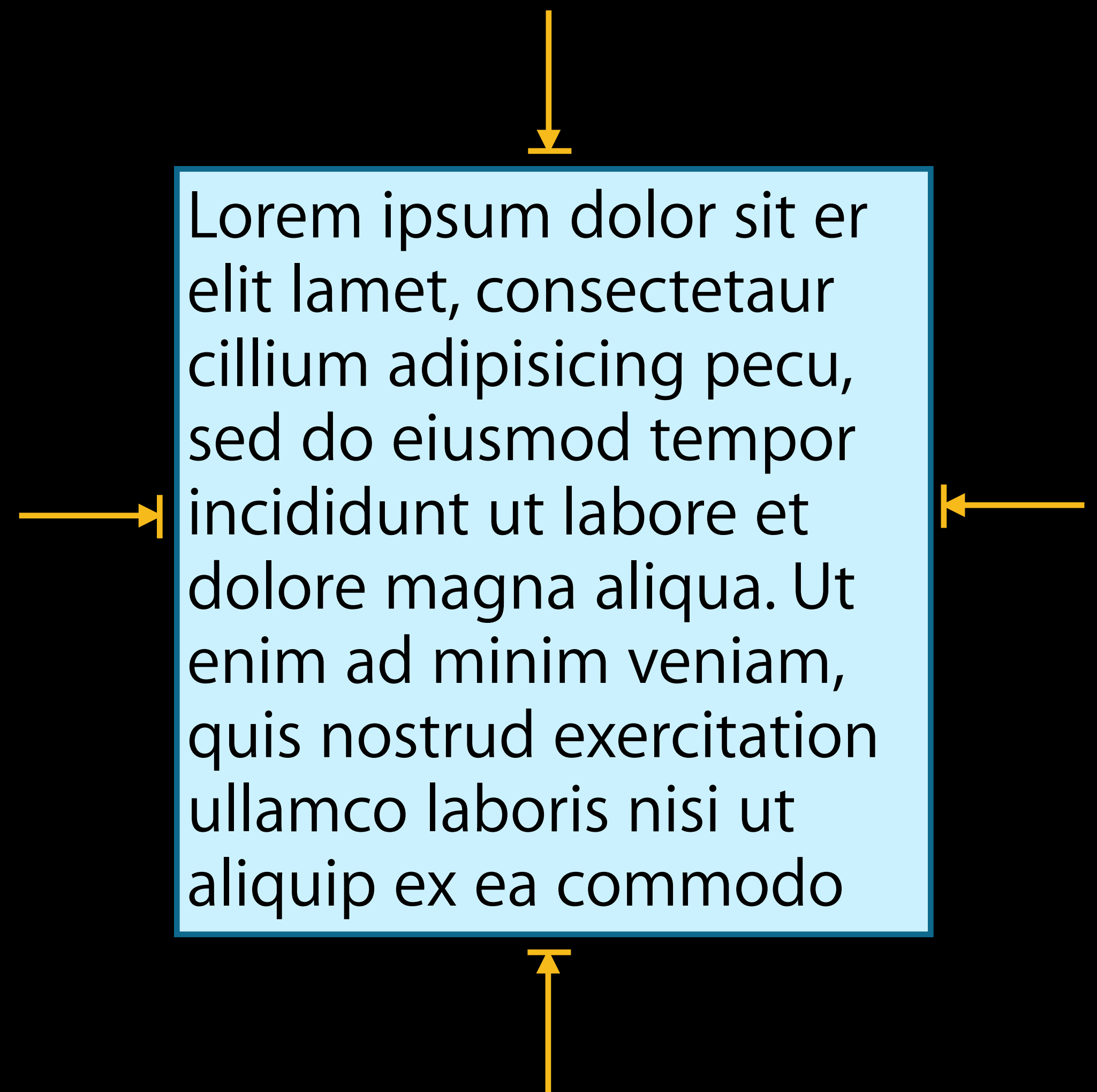
Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

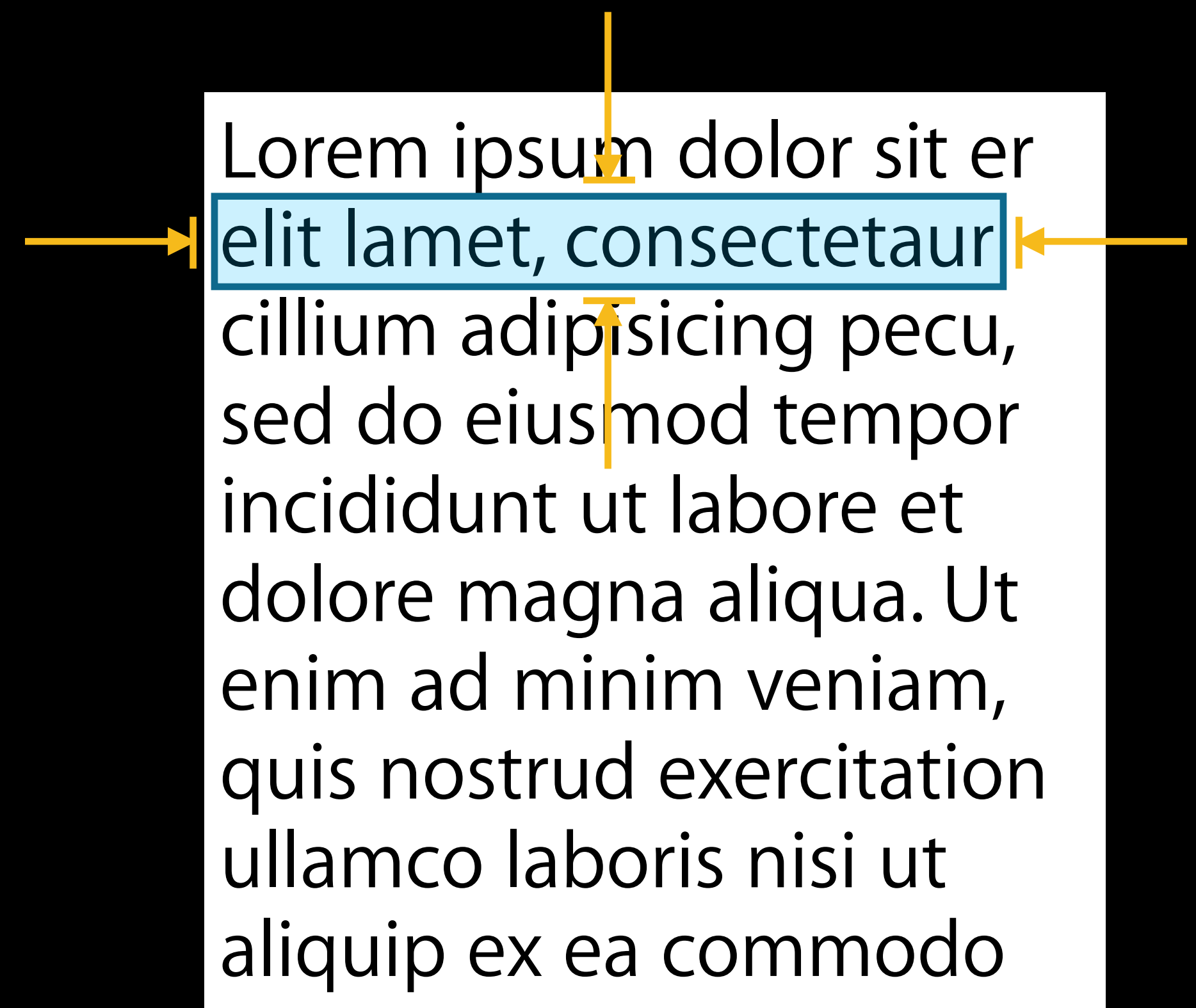
Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering



Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering



Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetaur
cillum adipisicing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. **Ut**
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum **adipiscing** pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetaur
cillum **adipisicing** pecu,
sed do eiusmod tempor
incididunt **ut** labore et
dolore **magna** aliqua. Ut
enim ad **minim** veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- **Hit-test touch location**
- Get precise location of a glyph
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, **consectetur**
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- **Get precise location of a glyph**
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipiscing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

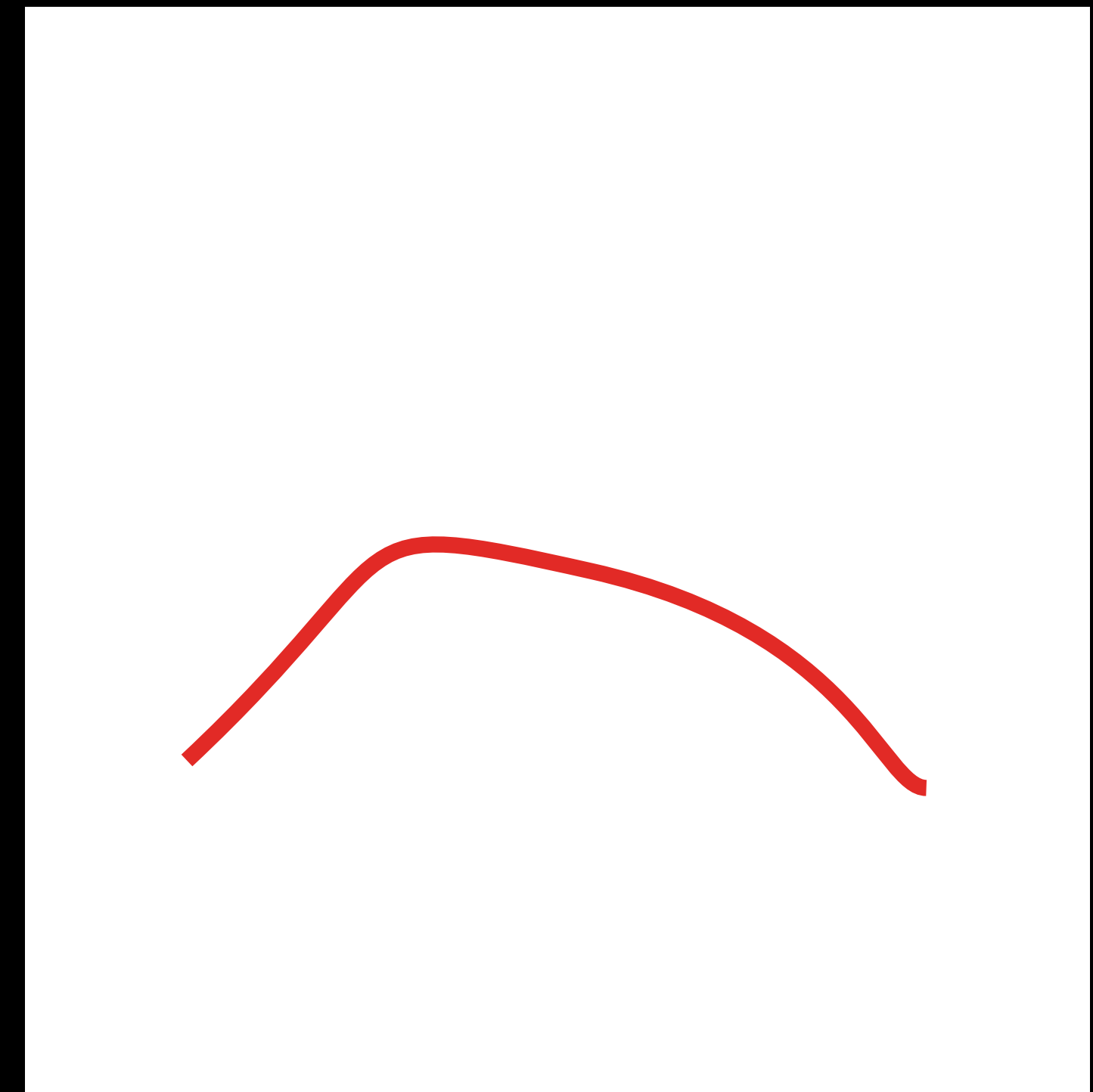
Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- **Get precise location of a glyph**
- Custom rendering

Lorem ipsum dolor sit er
elit lamet, consectetur
cillum adipis cing pecu,
sed do eiusmod tempor
incididunt ut labore et
dolore magna aliqua. Ut
enim ad minim veniam,
quis nostrud exercitation
ullamco laboris nisi ut
aliquip ex ea commodo

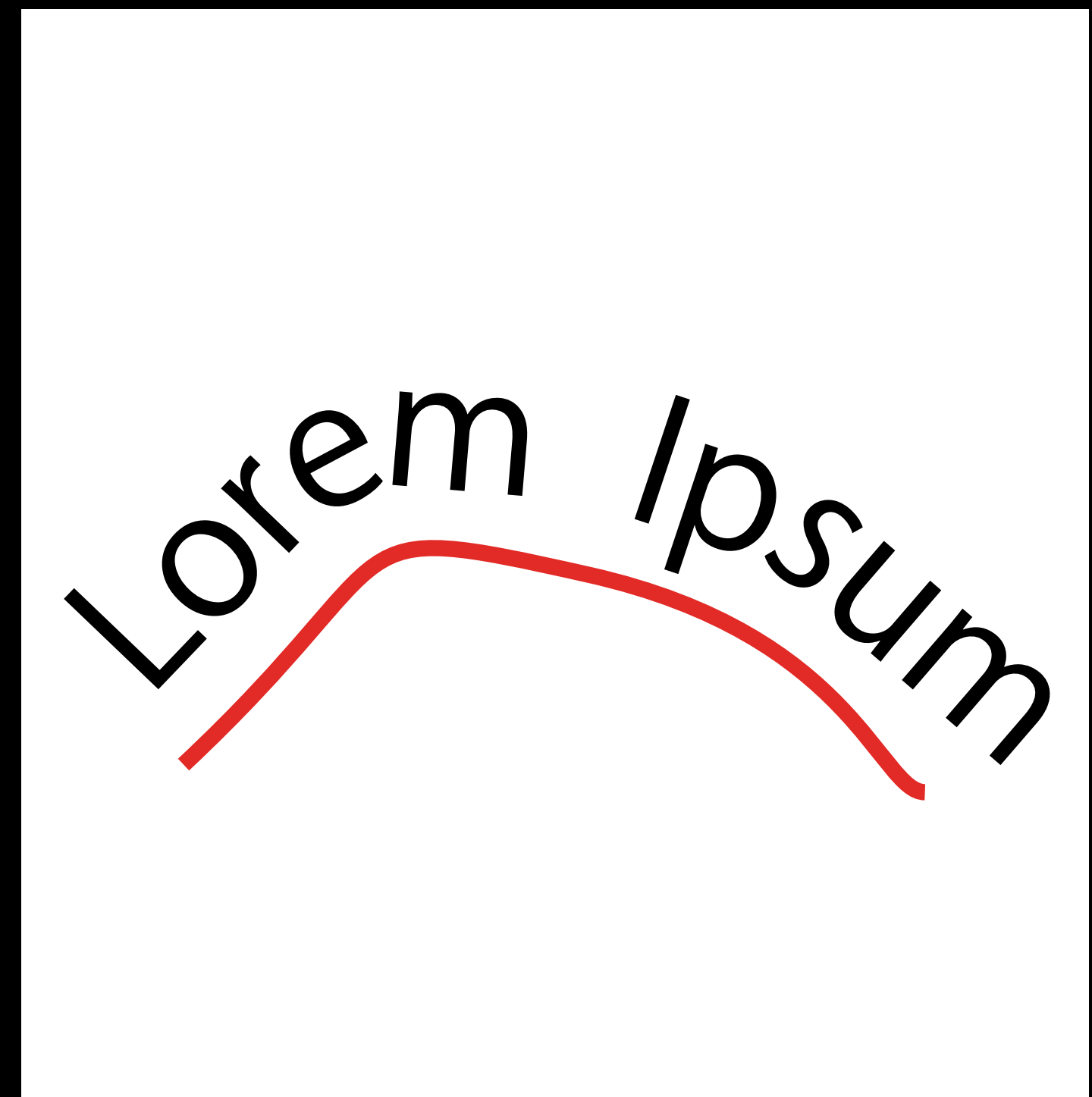
Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering



Working with Layout Information

- Measure text size down to a glyph
- Hit-test touch location
- Get precise location of a glyph
- Custom rendering



Glyph Information in NSLayoutManager

```
-(CGGlyph)glyphAtIndex:(NSUInteger)aGlyphIndex;
```

Glyph Information in NSLayoutManager

f

i

Glyph Information in NSLayoutManager

f

Glyph Information in NSLayoutManager

|Lorem ipsum dolor sit er elit lam|et, con

Glyph Information in NSLayoutManager

|Lorem ipsum dolor sit er elit lamet, con

Glyph Information in NSLayoutManager

The trouble-
makers

Glyph Information in NSLayoutManager

-(CGGlyph)glyphAtIndex:(NSUInteger)aGlyphIndex;

-(NSUInteger)characterIndexForGlyphAtIndex:(NSUInteger)aGlyphIndex;

Glyph Information in NSLayoutManager

-(CGGlyph)glyphAtIndex:(NSUInteger)aGlyphIndex;

-(NSUInteger)characterIndexForGlyphAtIndex:(NSUInteger)aGlyphIndex;

-(NSUInteger)glyphIndexForCharacterAtIndex:(NSUInteger)aCharIndex;

Text Layout Information

Text Layout

Text Layout Information

Helvetica Neue



Text Layout

Glyph ID: 59 76 95 91 51 72 96 86 92 91

Text Layout Information

Text Layout

Glyph ID: 59 76 95 91 51 72 96 86 92 91

Location: 0 4.2 8.4 12.1 16.1 20.9 25.2 29.8 34.8 38.1

Layout Information in NSLayoutManager

Layout Information in NSLayoutManager

- Text container

Layout Information in NSLayoutManager

- Text container
- Line

Layout Information in NSLayoutManager

- Text container
- Line
- Glyph location

Layout Information in NSLayoutManager

- Text container
- Line
- Glyph location

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

- Line
- Glyph location

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

- Line
- Glyph location

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

- Line
- Glyph location

Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt ut
labore et dolore magna aliqua.
Ut enim ad minim veniam, quis
nostrud exercitation ullamco
laboris nisi ut aliquip ex ea
commodo consequat. Duis aute
irure dolor in reprehenderit in

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```


- Line
- Glyph location

Lorem ipsum dolor sit er elit
amet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt ut
labore et dolore magna aliqua.
Ut enim ad minim veniam, quis
nostrud exercitation ullamco
laboris nisi ut aliquip ex ea
commodo consequat. Duis aute
irure dolor in reprehenderit in

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```


- Line
- Glyph location

Lorem ipsum dolor sit er elit		
amet,		consectetur
cillum		adipiscing
pecu,		sed do
eiusmod tempor incididunt ut		
labore et dolore magna aliqua.		
Ut enim ad minim veniam, quis		
nostrud exercitation ullamco		
laboris nisi ut aliquip ex ea		
commodo consequat. Duis aute		

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

- Line fragment
- Glyph location

Lorem ipsum dolor sit er elit		
amet,		consectetur
cillum		adipiscing
pecu,		sed do
eiusmod tempor incididunt ut		
labore et dolore magna aliqua.		
Ut enim ad minim veniam, quis		
nostrud exercitation ullamco		
laboris nisi ut aliquip ex ea		
commodo consequat. Duis aute		

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

```
-(CGRect)lineFragmentRectForGlyphAtIndex:(NSUInteger)aGlyphIndex  
effectiveRange:(NSRangePointer)aRangeP;
```

- Glyph location

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

```
-(CGRect)lineFragmentRectForGlyphAtIndex:(NSUInteger)aGlyphIndex  
effectiveRange:(NSRangePointer)aRangeP;
```

- Glyph location

Layout Information in NSLayoutManager

```
-(NSTextContainer *)textContainerForGlyphAtIndex:(NSUInteger)index  
effectiveRange:(NSRangePointer)aRangeP;
```

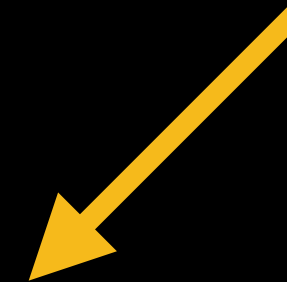
```
-(CGRect)lineFragmentRectForGlyphAtIndex:(NSUInteger)aGlyphIndex  
effectiveRange:(NSRangePointer)aRangeP;
```

```
-(CGPoint)locationForGlyphAtIndex:(NSUInteger)aGlyphIndex;
```

Text Layout Coordinate Systems

Text container coordinate

Text Container



Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incidunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud
exercitation ullamco laboris
nisi ut aliquip ex ea
commodo consequat. Duis

Text Layout Coordinate Systems

Text container coordinate

Text Container

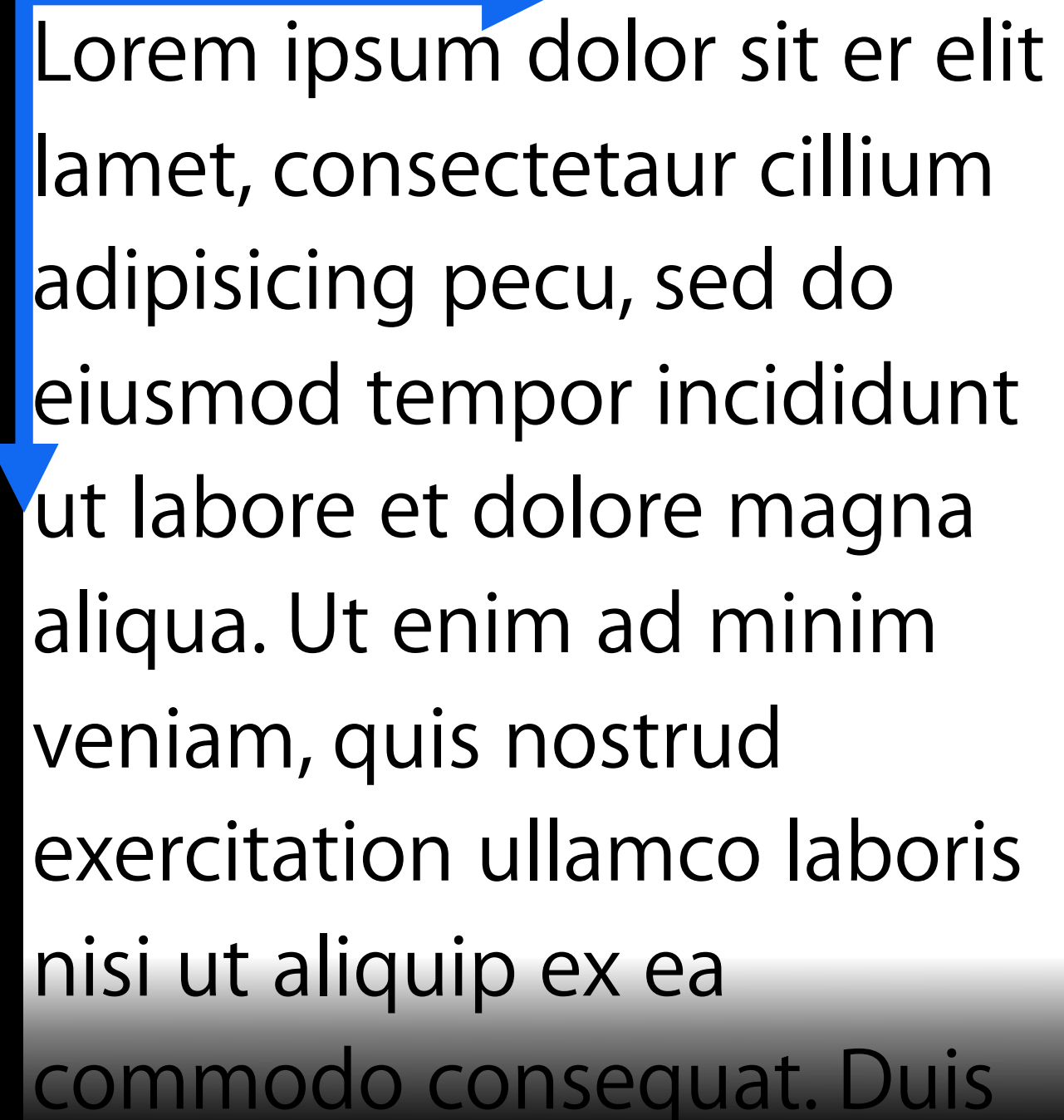
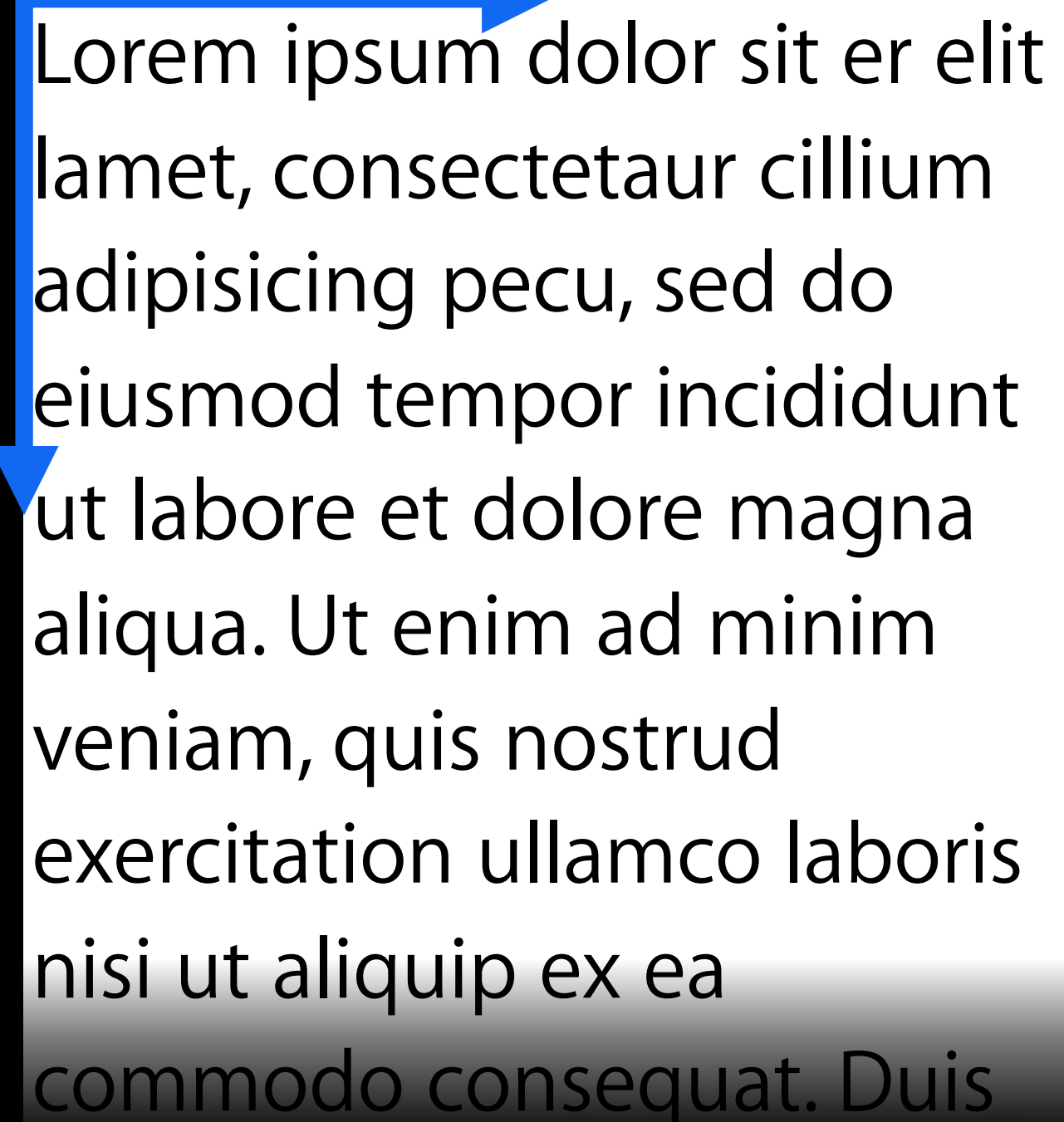


Diagram illustrating a text container coordinate system. A white rectangular box contains the text: "Lorem ipsum dolor sit er elit lamet, consectetur cillum adipisicing pecu, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis". A blue L-shaped coordinate system is overlaid on the top-left corner of the box, with arrows pointing right and down. A yellow arrow points from the text "Text Container" to the top-right corner of the box.

Lorem ipsum dolor sit er elit lamet, consectetur cillum adipisicing pecu, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis

Text Layout Coordinate Systems

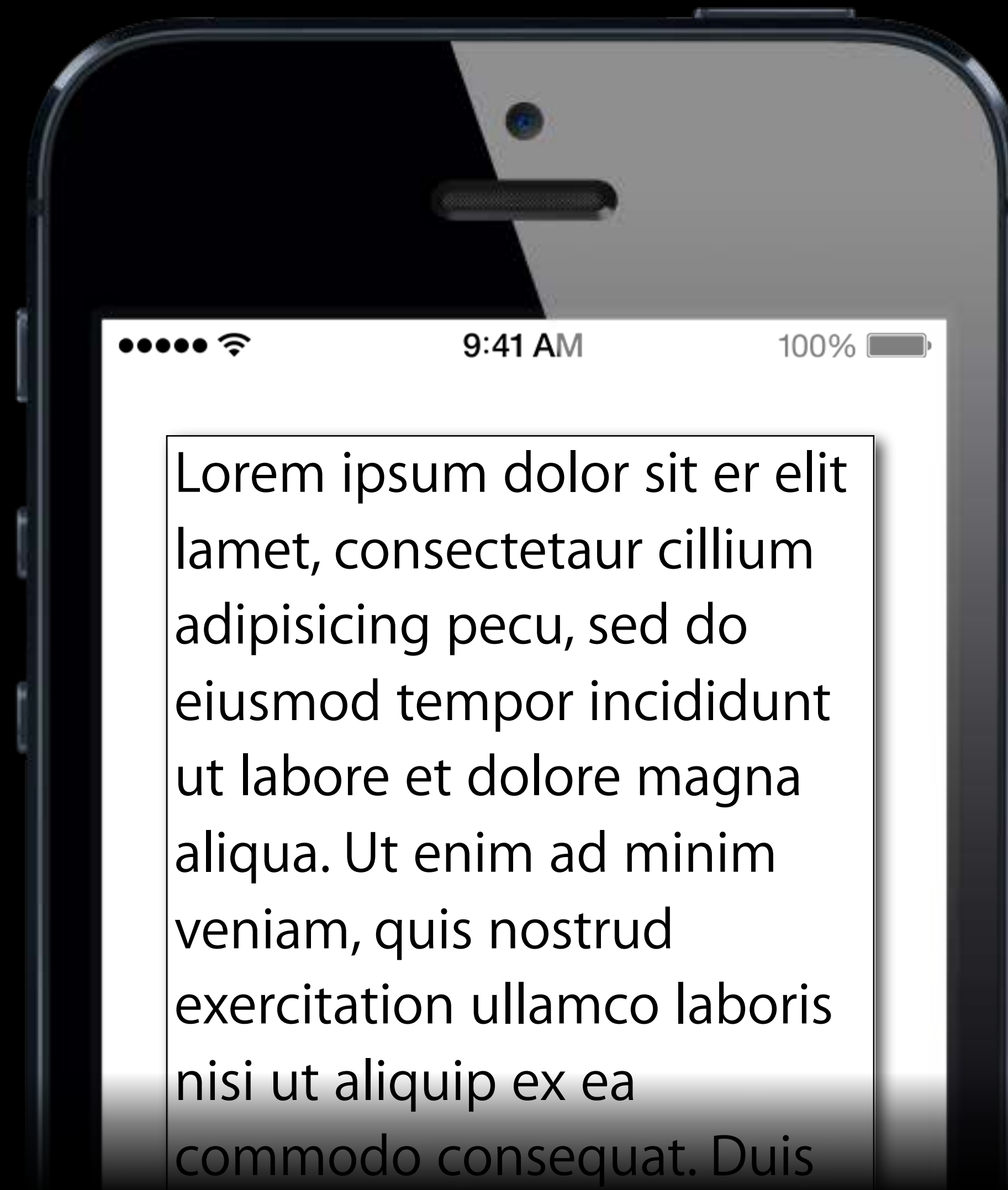
Text container coordinate



Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud
exercitation ullamco laboris
nisi ut aliquip ex ea
commodo consequat. Duis

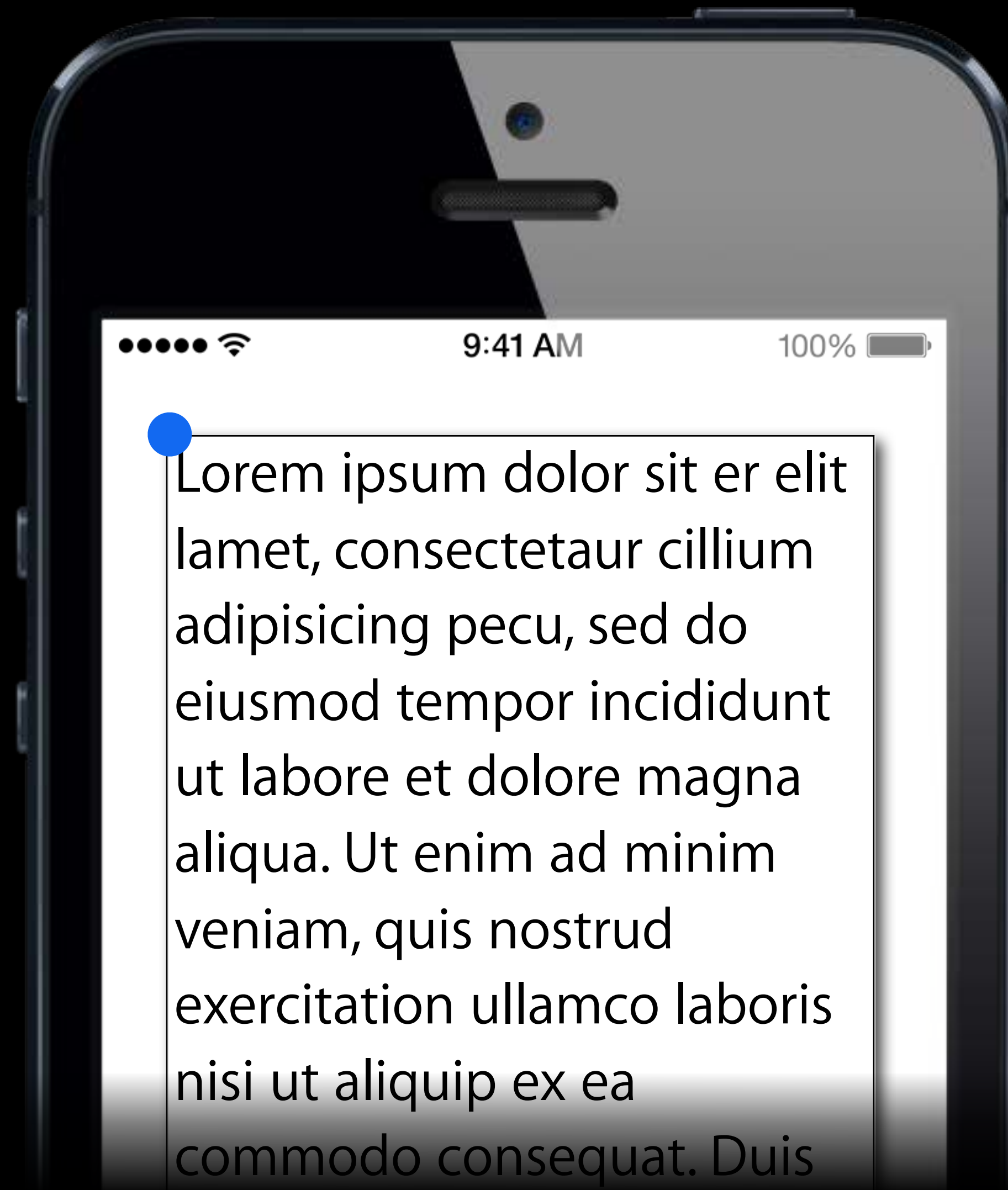
Text Layout Coordinate Systems

Text container coordinate



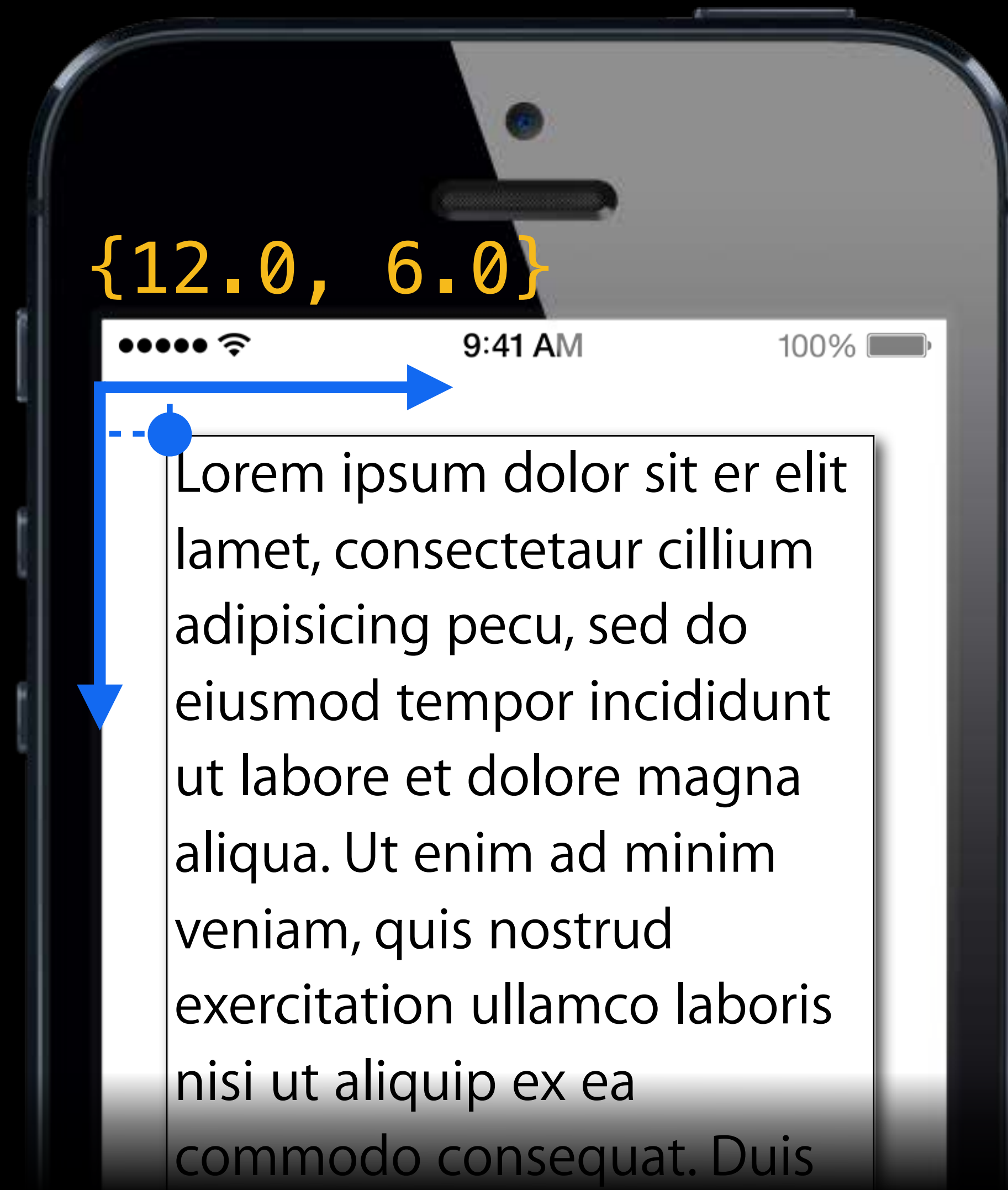
Text Layout Coordinate Systems

Text container coordinate



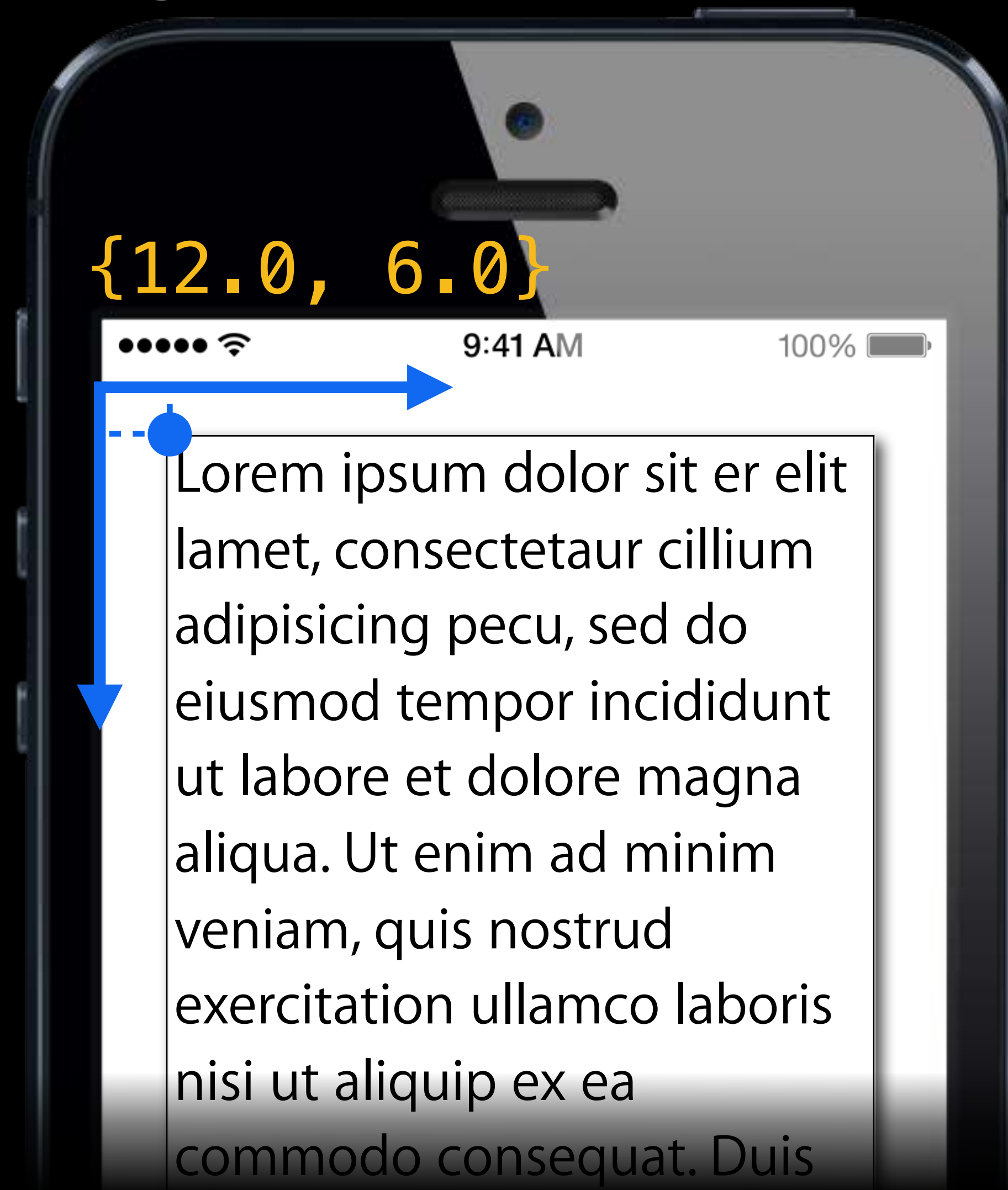
Text Layout Coordinate Systems

Text container coordinate



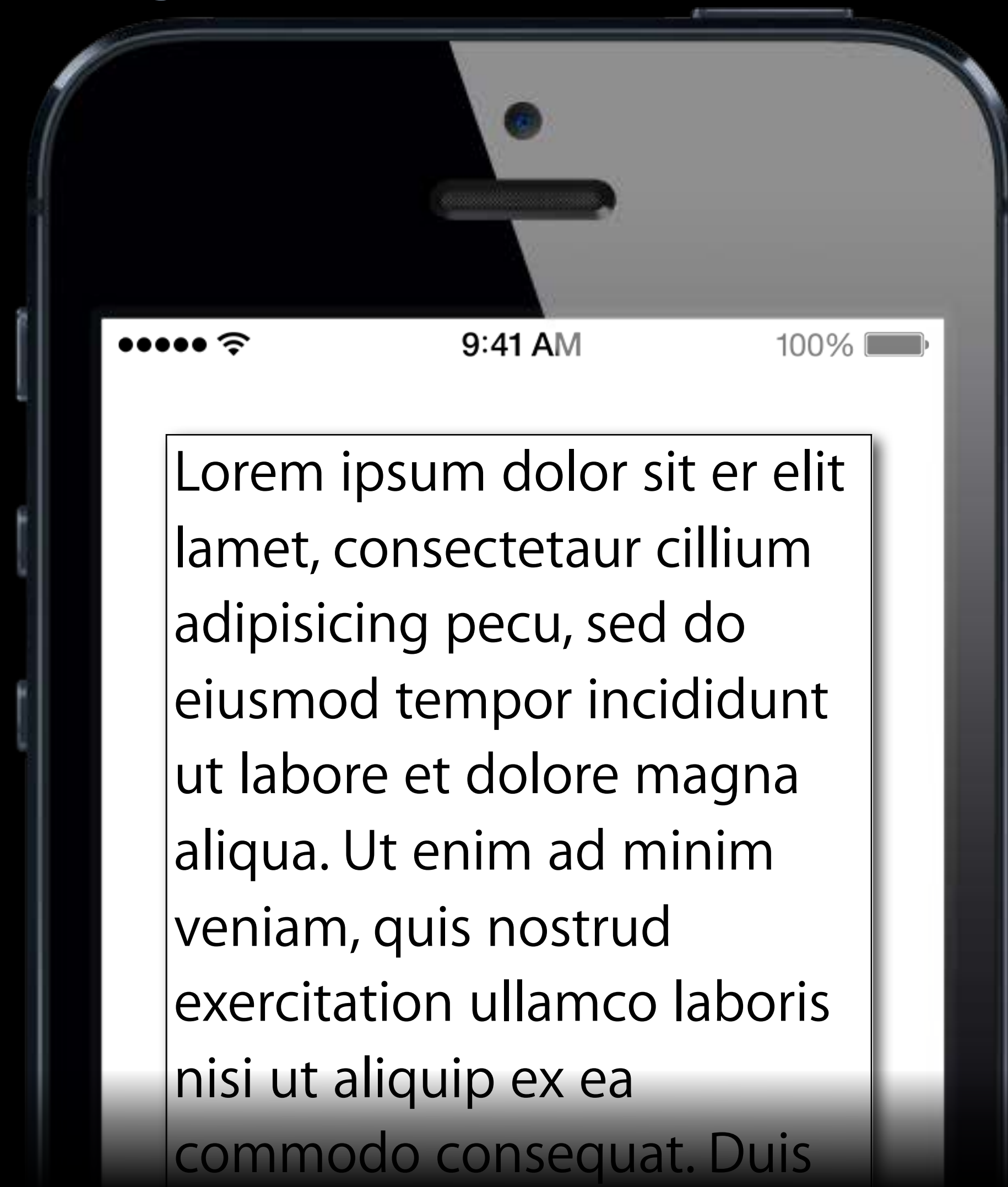
Text Layout Coordinate Systems

Line fragment rect origin




Text Layout Coordinate Systems

Line fragment rect origin



Text Layout Coordinate Systems

Line fragment rect origin

A smartphone is shown from a top-down perspective, with a white rectangular text box overlaid on its screen. The text box contains several lines of Lorem Ipsum placeholder text.

Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud

Text Layout Coordinate Systems

Line fragment rect origin

Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud

Text Layout Coordinate Systems

Line fragment rect origin

$\{0.0, 68.0\}$

>Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud

Line Fragment

Text Layout Coordinate Systems

Line fragment rect coordinate

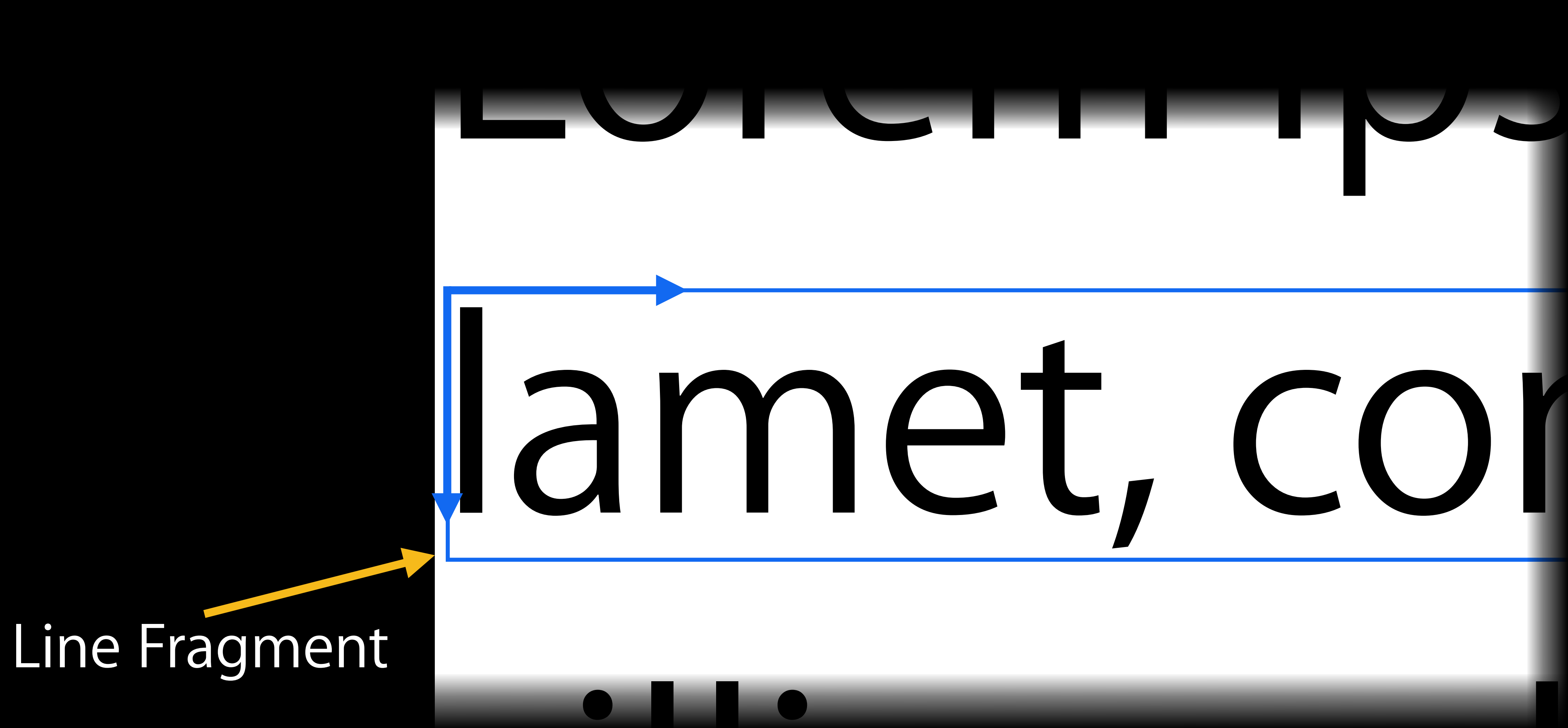
LOREUM IPSUM

amet, con

• | •

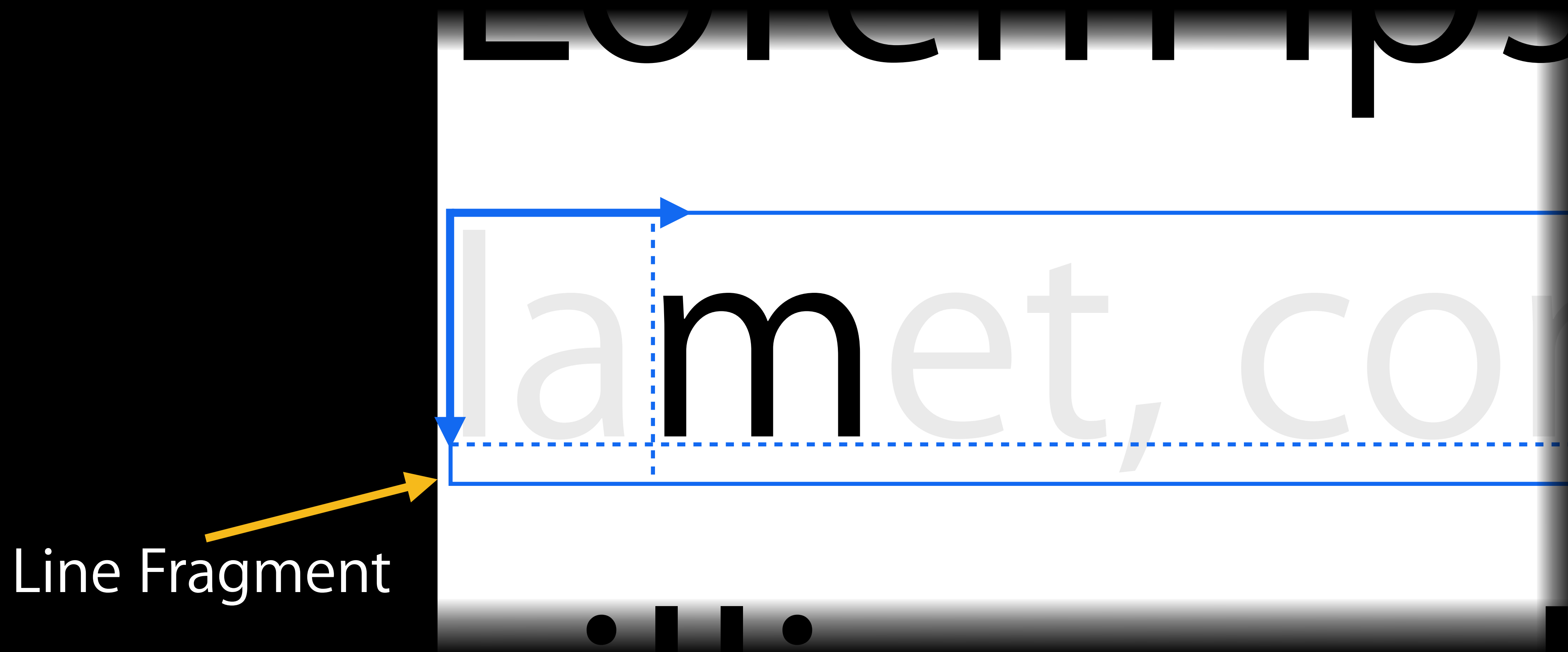
Text Layout Coordinate Systems

Line fragment rect coordinate



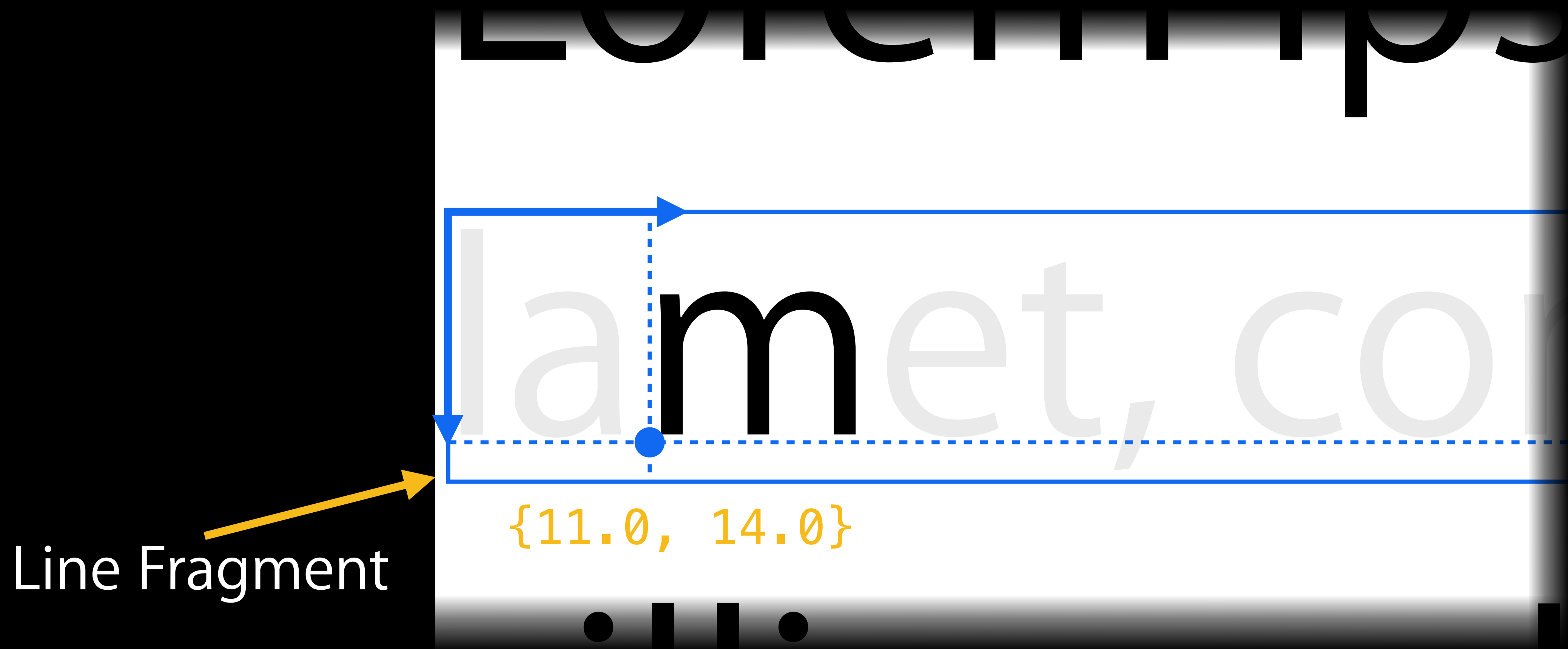
Text Layout Coordinate Systems

Line fragment rect coordinate



Text Layout Coordinate Systems

Line fragment rect coordinate



Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSUInteger characterIndex = textView.textStorage.length - 1;
NSUInteger glyphIndex;
CGRect rect;
CGPoint glyphLocation;

glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];

rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex
                                     effectiveRange:NULL];

glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];

glyphLocation.x += CGRectGetMinX(rect);
glyphLocation.y += CGRectGetMinY(rect);
```

Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSUInteger characterIndex = textView.textStorage.length - 1;
NSUInteger glyphIndex;
CGRect rect;
CGPoint glyphLocation;

glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];

rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex
                                         effectiveRange:NULL];

glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];

glyphLocation.x += CGRectGetMinX(rect);
glyphLocation.y += CGRectGetMinY(rect);
```

Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;  
NSUInteger characterIndex = textView.textStorage.length - 1;  
NSUInteger glyphIndex;  
CGRect rect;  
CGPoint glyphLocation;
```

```
glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];
```

```
rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex  
                                         effectiveRange:NULL];
```

```
glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];
```

```
glyphLocation.x += CGRectGetMinX(rect);  
glyphLocation.y += CGRectGetMinY(rect);
```

Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSUInteger characterIndex = textView.textStorage.length - 1;
NSUInteger glyphIndex;
CGRect rect;
CGPoint glyphLocation;

glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];

rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex
        effectiveRange:NULL];

glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];

glyphLocation.x += CGRectGetMinX(rect);
glyphLocation.y += CGRectGetMinY(rect);
```

Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSUInteger characterIndex = textView.textStorage.length - 1;
NSUInteger glyphIndex;
CGRect rect;
CGPoint glyphLocation;

glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];

rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex
                                     effectiveRange:NULL];

glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];

glyphLocation.x += CGRectGetMinX(rect);
glyphLocation.y += CGRectGetMinY(rect);
```


Glyph Location inside UITextView

Finding the last character

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSUInteger characterIndex = textView.textStorage.length - 1;
NSUInteger glyphIndex;
CGRect rect;
CGPoint glyphLocation;

glyphIndex = [layoutManager glyphIndexForCharacterIndex:characterIndex];

rect = [layoutManager lineFragmentRectForGlyphAtIndex:glyphIndex
                                     effectiveRange:NULL];

glyphLocation = [layoutManager locationForGlyphAtIndex:glyphIndex];

glyphLocation.x += CGRectGetMinX(rect);
glyphLocation.y += CGRectGetMinY(rect);
```

Hit Testing

Finding the word under a touch

```
NSLayoutManager *layoutManager = textView.layoutManager;
CGPoint location = [touch locationInView:textView];
NSUInteger characterIndex;

characterIndex = [layoutManager characterIndexForPoint:location
                inTextContainer:textView.textContainer
                fractionOfDistanceBetweenInsertionPoints:NULL];

if (characterIndex < textView.textStorage.length) { // valid index
    // Find the word range here
    // using -enumerateSubstringsInRange:options:usingBlock:
}
```

Hit Testing

Finding the word under a touch

```
NSLayoutManager *layoutManager = textView.layoutManager;
CGPoint location = [touch locationInView:textView];
NSUInteger characterIndex;

characterIndex = [layoutManager characterIndexForPoint:location
                inTextContainer:textView.textContainer
                fractionOfDistanceBetweenInsertionPoints:NULL];

if (characterIndex < textView.textStorage.length) { // valid index
    // Find the word range here
    // using -enumerateSubstringsInRange:options:usingBlock:
}
```

Hit Testing

Finding the word under a touch

```
NSLayoutManager *layoutManager = textView.layoutManager;  
CGPoint location = [touch locationInView:textView];  
NSUInteger characterIndex;
```

```
characterIndex = [layoutManager characterIndexForPoint:location  
                 inTextContainer:textView.textContainer  
                 fractionOfDistanceBetweenInsertionPoints:NULL];
```

```
if (characterIndex < textView.textStorage.length) { // valid index  
    // Find the word range here  
    // using -enumerateSubstringsInRange:options:usingBlock:  
}
```

Hit Testing

Finding the word under a touch

```
NSLayoutManager *layoutManager = textView.layoutManager;
CGPoint location = [touch locationInView:textView];
NSUInteger characterIndex;

characterIndex = [layoutManager characterIndexForPoint:location
                inTextContainer:textView.textContainer
                fractionOfDistanceBetweenInsertionPoints:NULL];

if (characterIndex < textView.textStorage.length) { // valid index
    // Find the word range here
    // using -enumerateSubstringsInRange:options:usingBlock:
}
```


Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```

Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```


Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```

Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```

Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;  
CGRect renderingArea = ... ;  
CGPoint containerOrigin = ... ;  
NSRange glyphRange;
```

```
renderingArea.origin.x -= containerOrigin.x;  
renderingArea.origin.y -= containerOrigin.y;
```

```
glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea  
                inTextContainer:textView.textContainer];
```

```
if (glyphRange.length) {  
    [layoutManager drawBackgroundForGlyphRange:glyphRange  
                    atPoint:containerOrigin];  
    [layoutManager drawGlyphsForGlyphRange:glyphRange  
                    atPoint:containerOrigin];  
}
```

Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
```

```
CGRect renderingArea = ... ;
```

```
CGPoint containerOrigin = ... ;
```

```
NSRange glyphRange;
```

```
renderingArea.origin.x -= containerOrigin.x;
```

```
renderingArea.origin.y -= containerOrigin.y;
```

```
glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea  
                inTextContainer:textView.textContainer];
```

```
if (glyphRange.length) {
```

```
    [layoutManager drawBackgroundForGlyphRange:glyphRange  
                    atPoint:containerOrigin];
```

```
    [layoutManager drawGlyphsForGlyphRange:glyphRange  
                    atPoint:containerOrigin];
```

```
}
```

Rendering Glyphs

Manually rendering a part of text container

```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```

Rendering Glyphs

Manually rendering a part of text container

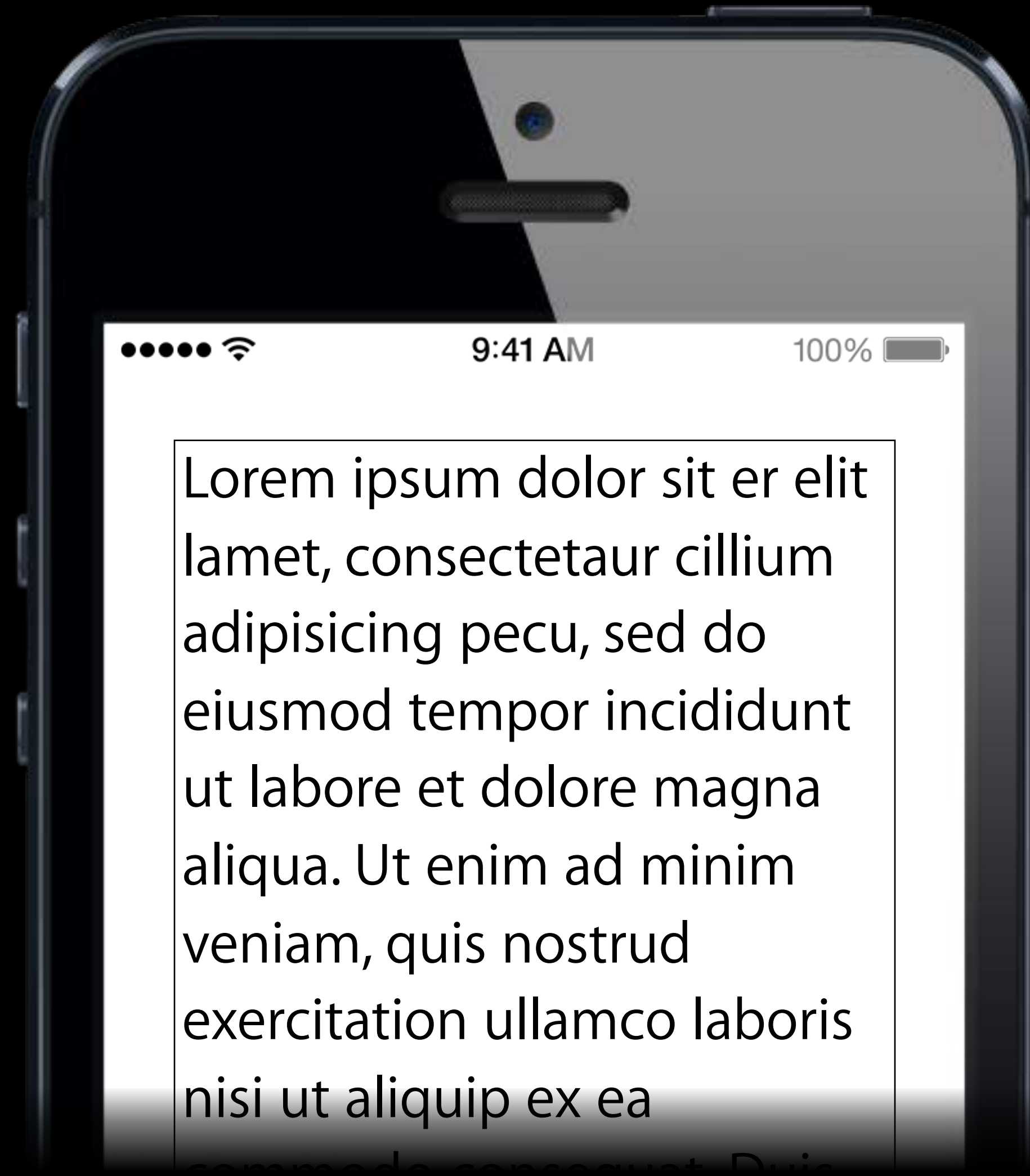
```
NSLayoutManager *layoutManager = ... ;
CGRect renderingArea = ... ;
CGPoint containerOrigin = ... ;
NSRange glyphRange;

renderingArea.origin.x -= containerOrigin.x;
renderingArea.origin.y -= containerOrigin.y;

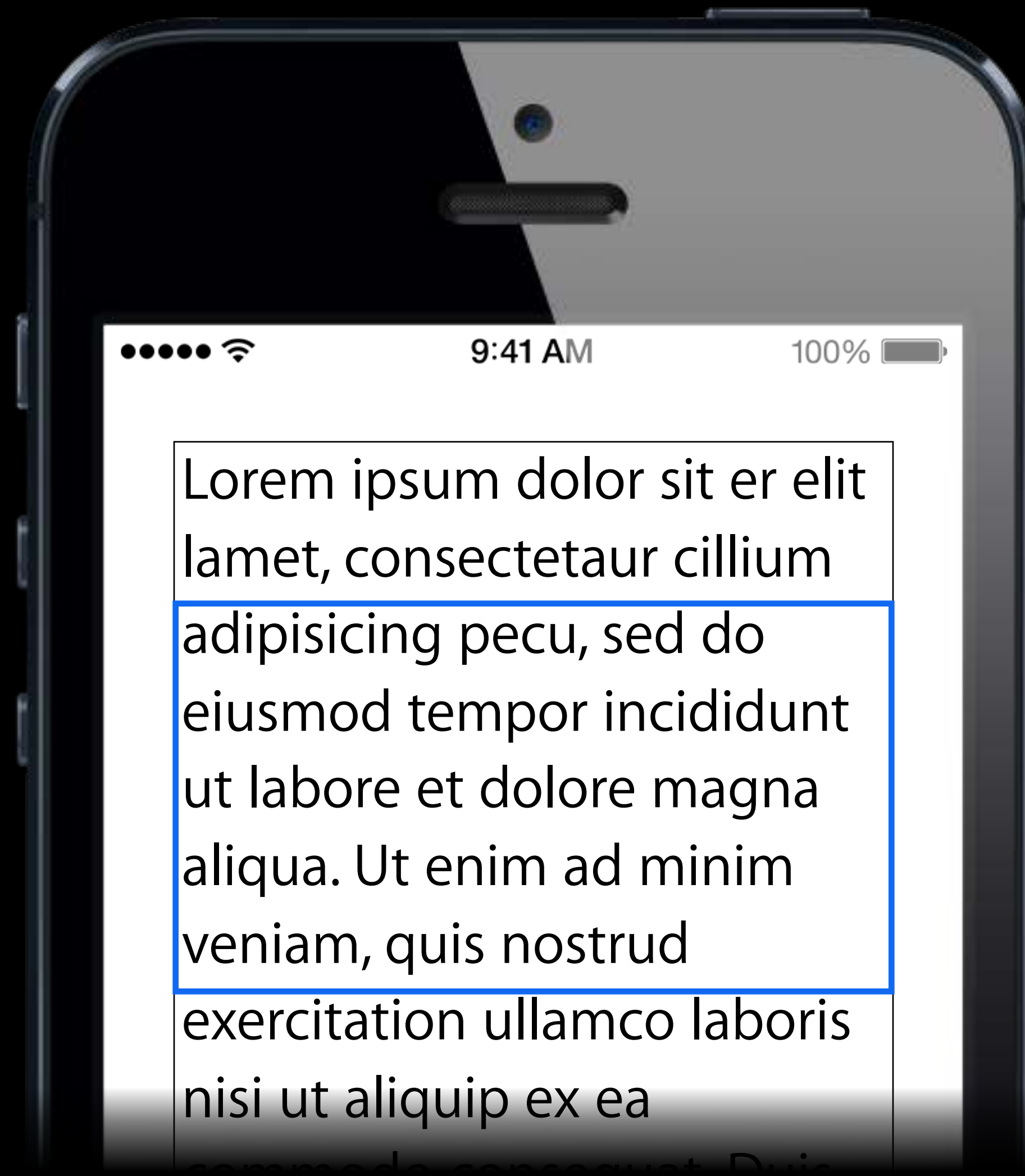
glyphRange = [layoutManager glyphRangeForBoundingRect:renderingArea
                    inTextContainer:textView.textContainer];

if (glyphRange.length) {
    [layoutManager drawBackgroundForGlyphRange:glyphRange
                    atPoint:containerOrigin];
    [layoutManager drawGlyphsForGlyphRange:glyphRange
                    atPoint:containerOrigin];
}
```

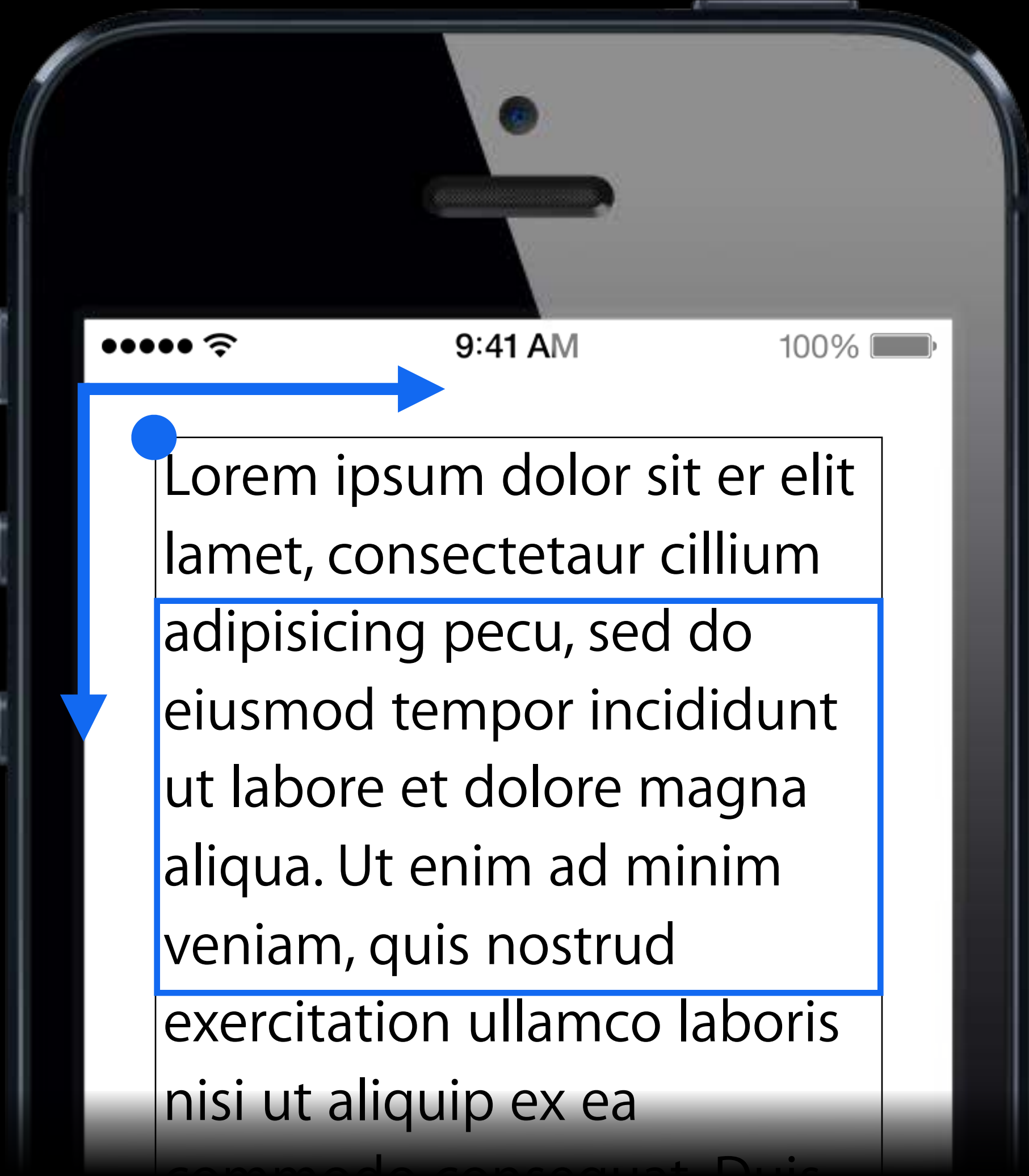

Location for Drawing Methods



Location for Drawing Methods

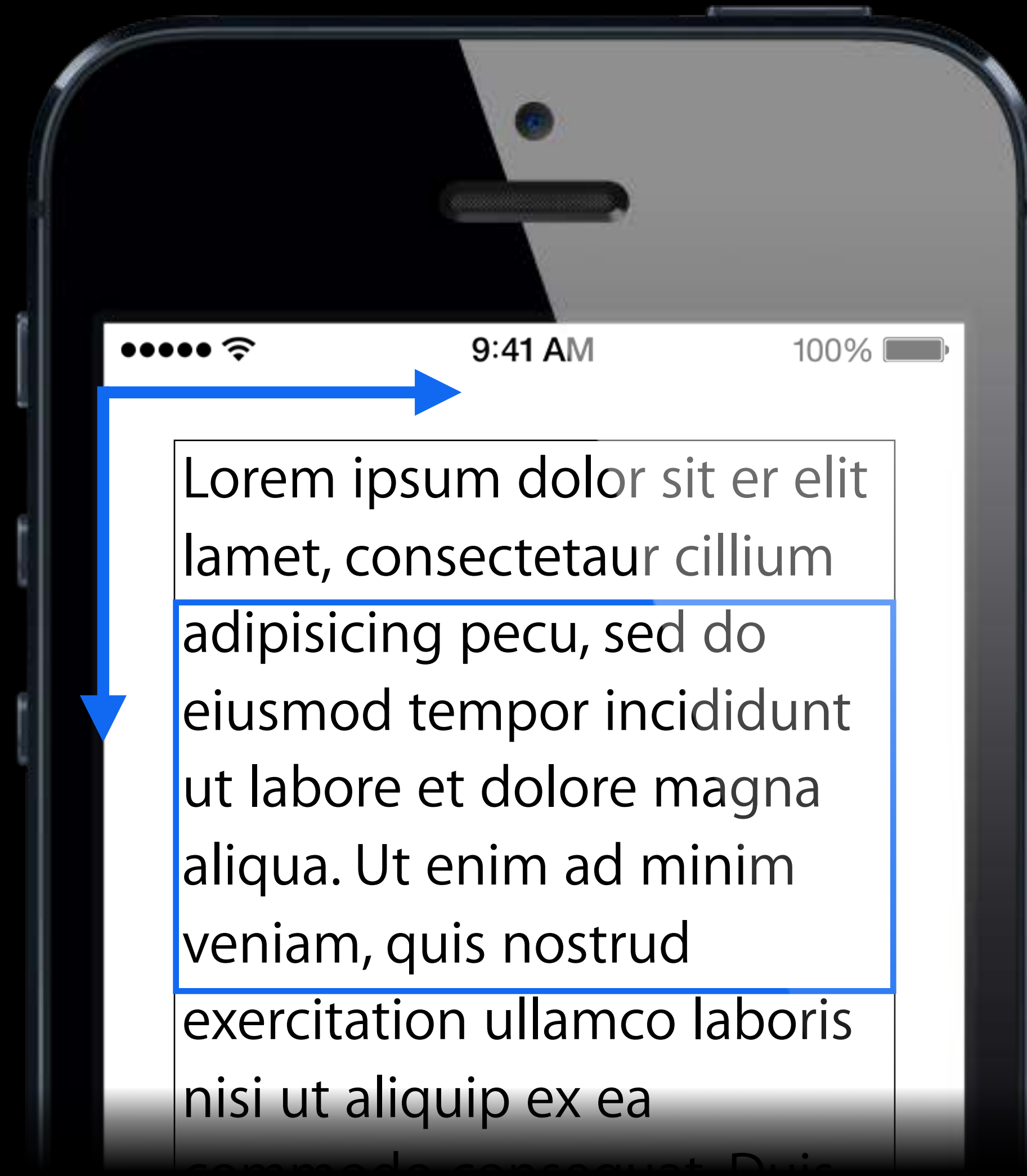


Location for Drawing Methods

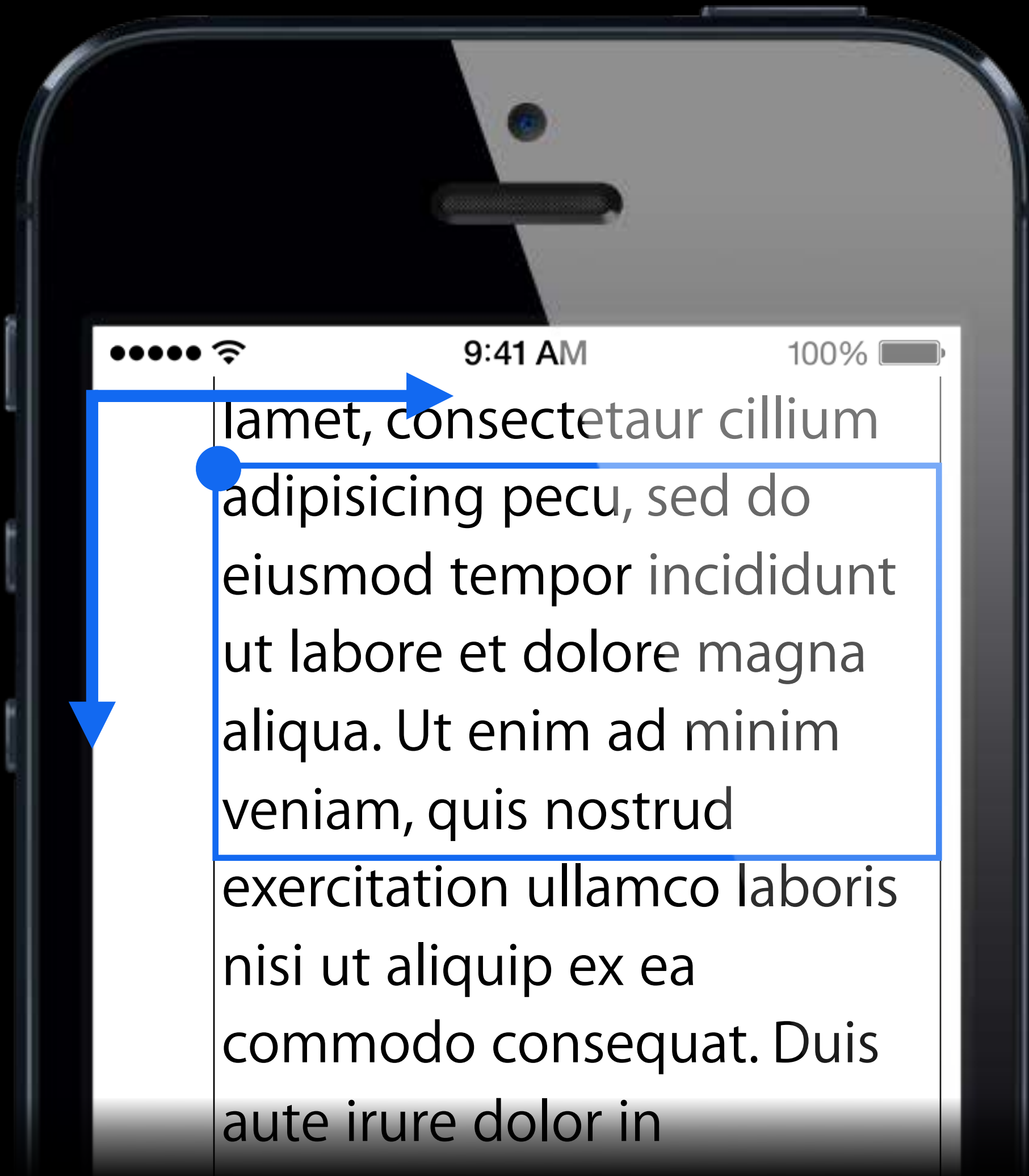


Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do
eiusmod tempor incididunt
ut labore et dolore magna
aliqua. Ut enim ad minim
veniam, quis nostrud
exercitation ullamco laboris
nisi ut aliquip ex ea

Location for Drawing Methods



Location for Drawing Methods



The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;

glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];

while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```

The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;

glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];

while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```

The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;
```

```
glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];
```

```
while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```

The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;

glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];

while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```

The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;

glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];

while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```

The Number of Lines

Counting the number of visual lines

```
NSLayoutManager *layoutManager = textView.layoutManager;
NSTextContainer *textContainer = textView.textContainer;
NSRange glyphRange, lineRange = NSMakeRange(0, 0);
NSRect rect;
CGFloat lastOriginY = -1.0;
NSUInteger numberOfLines = 0;

glyphRange = [layoutManager glyphRangeForTextContainer:textContainer];

while (lineRange.location < NSMaxRange(glyphRange)) {
    rect = [layoutManager lineFragmentRectForGlyphAtIndex:lineRange.location
                                             effectiveRange:&lineRange];
    if (CGRectMinY(rect) > lastOriginY) ++numberOfLines;
    lastOriginY = CGRectGetMinY(rect);
    lineRange.location = NSMaxRange(lineRange);
}
```


Customizing Text Layout

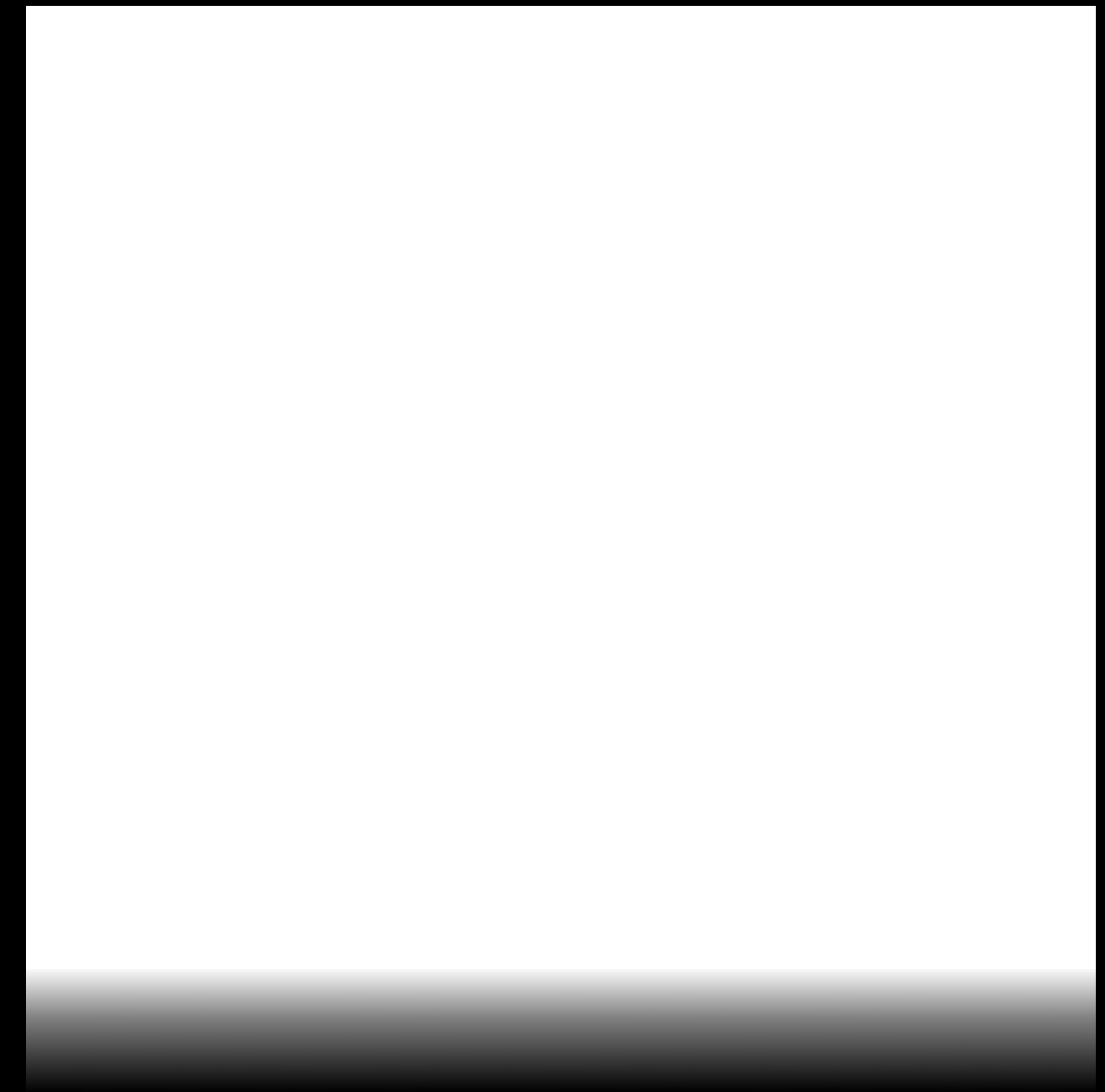
Customizing Text Layout

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing



Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing

Lorem ipsum dolor sit er elit
lamet, consectetur cillum
adipiscing pecu, sed do

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing

Lorem ipsum dolor sit er elit
lamet, consectetaur cillium
adipisicing pecu, sed do

eiusmod tempor incididunt
ut labore et dolore magna

aliqua. Ut enim ad minim
veniam, quis nostrud

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing

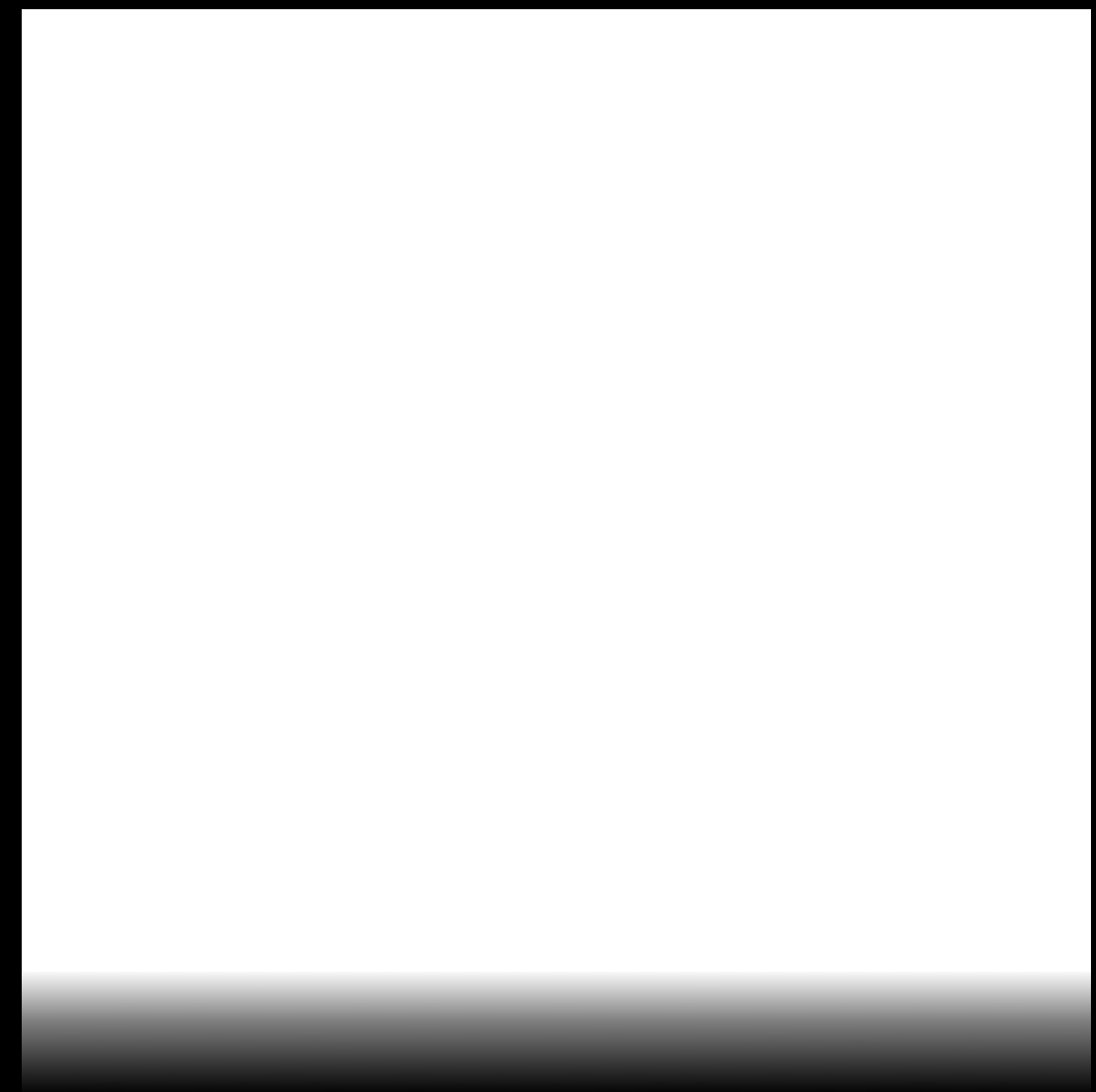
```
- (void)viewDidLoad {  
    self.enabled = true;  
    ! Property 'enabled' not found on object of type  
}
```

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking



Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking

Lorem ipsum dolor sit er elit
lamet, consectetaur cillium
adipisicing pecu, sed do



Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking

Lorem ipsum dolor sit er elit
lamet, consectetaur cillium
adipiscing pecu, sed do



Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking

Lorem ipsum dolor sit er elit
lamet, consectetaur cillium
adipisicing pecu, sed do
sed do eiusmod tempor

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping



Password

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping

Password


Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping

Heading 1

This is a snippet of text we'd like to hide.

Heading 2

This is a range of text we'd like to keep visible.

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping

Heading 1

Heading 2

This is a range of text we'd like to keep visible.

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping
- Subclassing NSTextAttachment
- Subclassing NSTextContainer

Customizing Text Layout

- NSLayoutManager delegate
 - Modifying line spacing
 - Validating soft line breaking
 - Custom glyph mapping
- Subclassing NSTextAttachment
- Subclassing NSTextContainer

Multiple Truncations

Truncating with a focused text

| Lorem ipsum dolor sit er elit lamet, consect... |

Multiple Truncations

Truncating with a focused text

| Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating with a focused text

| Lorem ipsum dolor sit er elit...consectetur... |

Multiple Truncations

Truncating a range before the focused range

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

- truncatedGlyphRangeInLineFragmentForGlyphAtIndex:

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

-truncatedGlyphRangeInLineFragmentForGlyphAtIndex:

| Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

-truncatedGlyphRangeInLineFragmentForGlyphAtIndex:



>Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range
–truncatedGlyphRangeInLineFragmentForGlyphAtIndex:
2. Estimate the additional truncation range and re-layout

| Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range
–truncatedGlyphRangeInLineFragmentForGlyphAtIndex:
2. Estimate the additional truncation range and re-layout



Lorem ipsum dolor sit er elit lamet, **consectetaur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

–truncatedGlyphRangeInLineFragmentForGlyphAtIndex:

2. Estimate the additional truncation range and re-layout

3. Custom truncate in glyph generation

–layoutManager:shouldGenerateGlyphs:

properties:characterIndexes:font:forGlyphRange:

| Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

`-truncatedGlyphRangeInLineFragmentForGlyphAtIndex:`

2. Estimate the additional truncation range and re-layout

3. Custom truncate in glyph generation

`-layoutManager:shouldGenerateGlyphs:`

`properties:characterIndexes:font:forGlyphRange:`

| Lorem ipsum dolor sit er elit lamet, **consectetur** cill

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

`-truncatedGlyphRangeInLineFragmentForGlyphAtIndex:`

2. Estimate the additional truncation range and re-layout

3. Custom truncate in glyph generation

`-layoutManager:shouldGenerateGlyphs:`

`properties:characterIndexes:font:forGlyphRange:`

| Lorem ipsum dolor sit er elit...consectetur... |

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range
 - truncatedGlyphRangeInLineFragmentForGlyphAtIndex:
2. Estimate the additional truncation range and re-layout
3. Custom truncate in glyph generation
 - layoutManager:shouldGenerateGlyphs:
 - properties:characterIndexes:font:forGlyphRange:
4. Go back to 1.

Multiple Truncations

Truncating a range before the focused range

1. Check if the truncated range intersects with the special range

–truncatedGlyphRangeInLineFragmentForGlyphAtIndex:

2. Estimate the additional truncation range and re-layout

3. Custom truncate in glyph generation

–layoutManager:shouldGenerateGlyphs:

properties:characterIndexes:font:forGlyphRange:

4. Go back to 1.

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) characterIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

Customizing Glyph Mappings

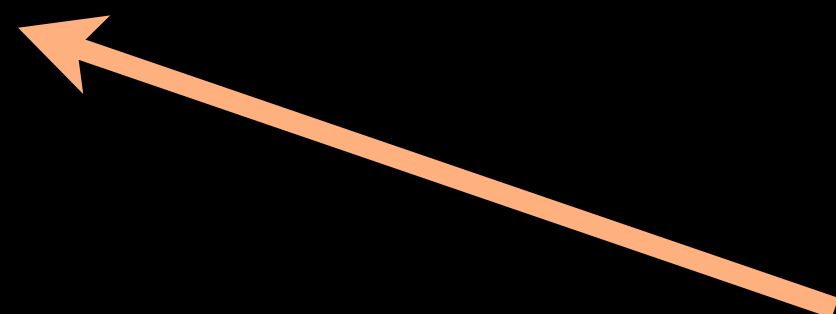
Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



delegate object

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

Lorem ipsum

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

Lorem ipsum

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

dolor sit er

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

dolor sit er

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

elit lamet,

Customizing Glyph Mappings

Substituting with the ellipsis


- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

elit lamet,

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) **charIndexes**
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

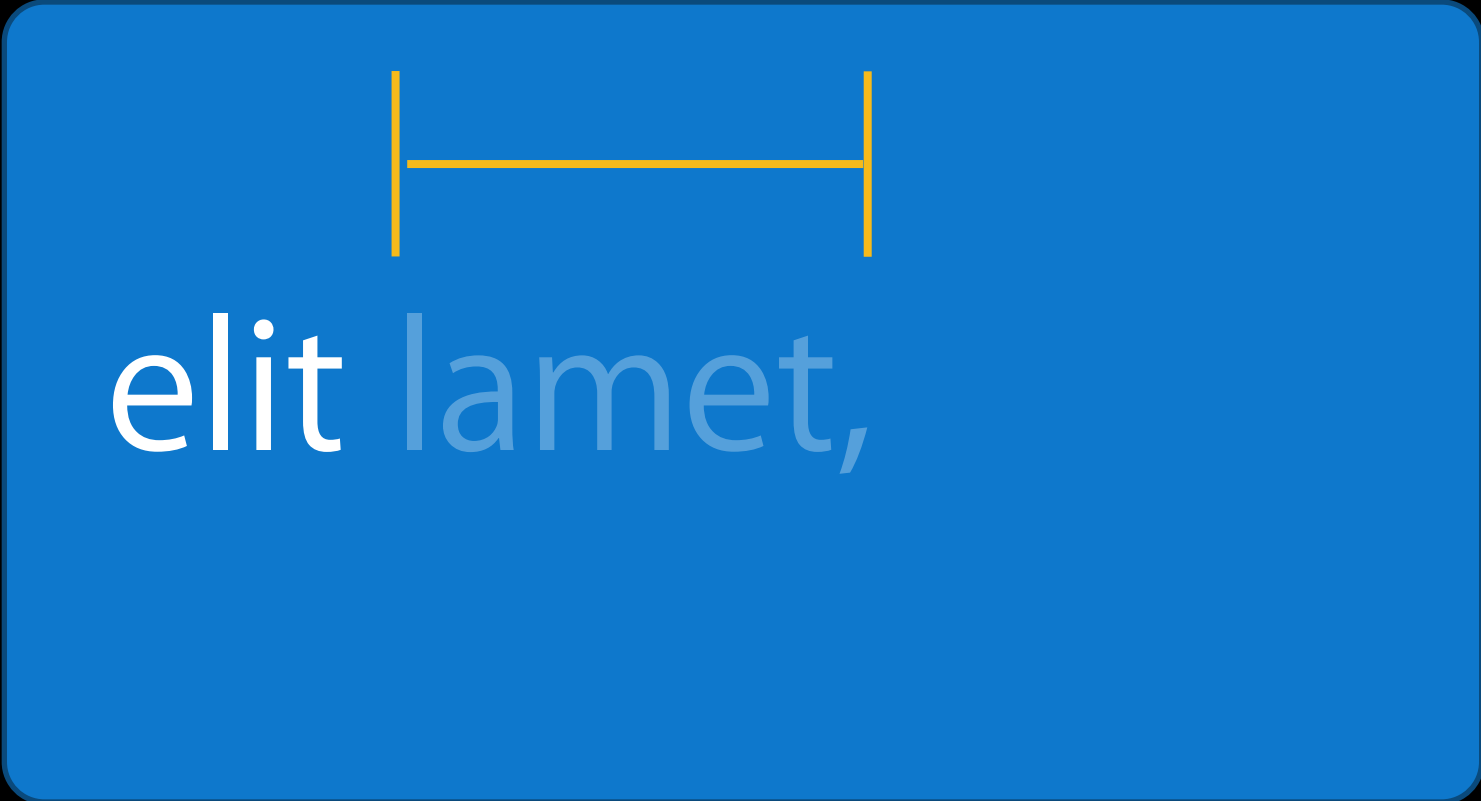


elit lamet,

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) **characterIndexes**
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



elit lamet,

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) **characterIndexes**
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



elit lamet...

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;

elit ...

Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Customizing Glyph Mappings

Substituting with the ellipsis

- (NSUInteger) layoutManager: (NSLayoutManager *) layoutManager
shouldGenerateGlyphs: (const CGGlyph *) glyphs
properties: (const NSGlyphProperty *) props
characterIndexes: (const NSUInteger *) charIndexes
font: (UIFont *) aFont forGlyphRange: (NSRange) glyphRange;



Glyph Properties

- (NSGlyphProperty)propertyForGlyphAtIndex:(NSUInteger)glyphIndex;

Glyph Properties

- (NSGlyphProperty)**propertyForGlyphAtIndex:(NSUInteger)glyphIndex;**
 - NSGlyphPropertyNull
 - NSGlyphPropertyControlCharacter
 - NSGlyphPropertyElastic
 - NSGlyphPropertyNonBaseCharacter

Glyph Properties

- (NSGlyphProperty)propertyForGlyphAtIndex:(NSUInteger)glyphIndex;
 - NSGlyphPropertyNull
 - NSGlyphPropertyControlCharacter
 - NSGlyphPropertyElastic
 - NSGlyphPropertyNonBaseCharacter

Demo

Multiple Truncations

Jordan Breeding
UIKit Engineer

Summary

Summary

- New letterpress text effect

Summary

- New letterpress text effect
- Flexible configurations with main Text Kit objects

Summary

- New letterpress text effect
- Flexible configurations with main Text Kit objects
- Readily accessible text layout information

Summary

- New letterpress text effect
- Flexible configurations with main Text Kit objects
- Readily accessible text layout information
- Open and many customization points

More Information

Jake Behrens

App Frameworks Evangelist
behrens@apple.com

Documentation

<http://developer.apple.com/library/ios/>

Apple Developer Forums

<http://devforums.apple.com>

Related Sessions

Introduction to Text Kit

Presidio
Wednesday 2:00PM

Using Fonts with Text Kit

Presidio
Friday 9:00AM

Labs

Text Kit and Core Text Lab

Frameworks Lab B
Wednesday 4:30PM

Text Kit and Core Text Lab

Frameworks Lab A
Thursday 4:30PM

 WWDC2013