Structure

AssignedWorkDocument

An assigned work document is a document that contains taker and scorer markup specific to a taker. It also contains a copy of the assignable document upon which it is based.

iOS 17.4+ | iPadOS 17.4+ | Mac Catalyst 17.4+ | visionOS

```
struct AssignedWorkDocument
```

Overview

This document has a collection of `AssignedWorkDocument.ScoreAnnotation` objects that represent marks such as correct and incorrect marks on a page. Score annotations are not automatically associated with a question defined in the `AssignableDocument` that this work document is based on. To determine the score for the work, you use `computeScore`.

You cannot instantiate this document type directly. Instead, you instantiate it by calling `assign(to:)` or `makeAssignedWorkDocument()`.

This document is fully mergeable, which means that any copies of this document that are independently mutated can be merged into a determinisitic resulting document. You can merge copies of this document into this one using `merge(_:)`. You can also merge individual parts of copies of this document into this one with `merge(partData:into:)`. For example, if deviceA has documentA and deviceB has documentB, which is a copy of documentA. When a user changes a question in documentB on deviceB, deviceB can export that part's data and send it to deviceA to be merged back into documentA.

You can create as many of these objects as you have memory for. This type assumes single-threaded access.

Topics

Creating an assigned work document

`init(id: AssignedWorkDocument.ID, assignableDocument: AssignableDocument, partData: [AssignedWorkDocument.PartID : MergeablePartData]) async throws`

Construct an instance of this object with the parts data passed in.

~~`init(id: AssignedWorkDocument.ID, assignableDocument: AssignableDocument, partData: [AssignedWorkDocument.PartID : URL]) throws`~~

Construct an instance of this object with the parts data passed in.

Deprecated

Inspecting a work document

`typealias ID`

A type representing the stable identity of this document.

`var id: AssignedWorkDocument.ID`

The stable identity of this document.

`var isMultiPageDocument: Bool`

`true`, if this document has more than one page; `false`, otherwise.

`var isPartial: Bool`

Denotes whether or not this document is a partial one.

`enum PartIDs`

An enumeration containing the identities of parts managed by this view.

`var partIDs: [MergeablePartsContainerPartID]`

Returns a collection of identifiers reflecting the manifest of parts available in the document.

`var scoreAnnotations: [AssignedWorkDocument.ScoreAnnotation]`

The collection of score annotations for this work document. Treated as a multiset. i.e. The order of the elements doesn't matter and duplicate values are allowed.

`var scorers: [AnyUserIdentity]`

The identities of users scoring this assigned work. Treated as a set.

```
var pagesDebugDescription: String
```

```
enum Error
```

Errors for this document type.

Getting the assignable document

```
var assignableDocument: AssignableDocument
```

The assignable document that this work document is based on.

Getting the assignees

```
var assignees: [AnyUserIdentity]
```

The identities of takers of this document. Treated as a set.

Getting the configuration

```
typealias Configuration
```

The configuration for an assessment taker work which contains an optional manual score for the document.

```
var configuration: any AssignedWorkDocumentConfiguration
```

The configuration for a taker work which contains an optional manual score for the document.

Merging the parts

```
func merge(AssignedWorkDocument) async throws -> Bool
```

Merge another object of this type into this object.

```
func merge(partData: MergeablePartData, into: AssignedWorkDocument.PartID) async throws -> Bool
```

Merges an individual part into the specified part of this object.

~~`func merge(other: AssignedWorkDocument) throws -> Bool`~~

Merge another object of this type into this object.

Deprecated

~~`func merge(partID: AssignedWorkDocument.PartID, partDataURL: URL) throws -> Bool`~~

Merges an individual part's data into the specified part of this object.

Deprecated

Producing thumbnails

`func questionThumbnails(visibleParts: [AssignedWorkDocument.PartID]) async -> [AssignableDocument.Question.ID : [AssignableDocument.Question.Thumbnail]]`

Produces thumbnails of question regions within the document.

Computing the score

`func computeScore() -> Double`

Gathers all of the points based on all the `AssignedWorkDocument.ScoreAnnotations` in the document and its `kind` property.

`struct ScoreAnnotation`

A score mark on page of the work document.

Making the parts

`func makePart(for: AssignedWorkDocument.PartID) throws -> MergeablePartData?`

Creates data for the part with the given identifier.

Exporting the parts

`func exportParts(identifiedBy: [AssignedWorkDocument.PartID]) async throws -> [AssignedWorkDocument.PartID : MergeablePartData]`

Given a set of part identifiers, return a dictionary of part ID to part data.

~~`func export(partIDs: [AssignedWorkDocument.PartID]) async throws -> [AssignedWorkDocument.PartID : URL]`~~

Given a set of part identifiers, return a dictionary of part ID to data objects for the requested parts.

Deprecated

Comparing work documents

```
static func == (AssignedWorkDocument, AssignedWorkDocument) -> Bool
```

Returns a Boolean value indicating whether two values are equal.

Hashing the work document

```
func hash(into: inout Hasher)
```

Hashes the essential components of this value by feeding them into the given hasher.

Accessing work documents

```
subscript(AssignedWorkDocument.Page.ID) -> AssignedWorkDocument.Page?
```

Access the page that the ID points to, if any.

```
subscript(AssignedWorkDocument.ScoreAnnotation.ID) -> AssignedWork
Document.ScoreAnnotation?
```

Access the score annotation that the identifier refers to, if any.

Default Implementations

MergeableDocument Implementations

Relationships

Conforms To

Copyable
Equatable
Hashable
Identifiable
MergeableDocument
MergeablePartsContainer

See Also

Assignable document

`struct AssignableDocument`

An assignable document is an augmented PDF that allows teachers to mark up the PDF with the intention of students taking the assessment.

`protocol Assignable`

Documents conforming to this protocol can be assigned to a user.