Class

LanguageModelSession

An object that represents a session that interacts with a language model.

iOS 26.0+ | iPadOS 26.0+ | Mac Catalyst 26.0+ | macOS 26.0+ | visionOS 26.0+

```
final class LanguageModelSession
```

Mentioned in

- Generating content and performing tasks with Foundation Models
- Categorizing and organizing data with content tags
- Generating Swift data structures with guided generation
- Improving the safety of generative model output
- Support languages and locales with Foundation Models

Overview

A session is a single context that you use to generate content with, and maintains state between requests. You can reuse the existing instance or create a new one each time you call the model. When creating a session, provide instructions that tells the model what its role is and provide guidance on how to respond.

```
let session = LanguageModelSession(instructions: """
    You are a motivational workout coach that provides quotes to inspire \
    and motivate athletes.
    """
)
let prompt = "Generate a motivational quote for my next workout."
```

```
let response = try await session.respond(to: prompt)
```

The framework records each call to the model in a `Transcript` that includes all prompts and responses. If your session exceeds the available context size, it throws `LanguageModelSession.GenerationError.exceededContextWindowSize(_:)`.

When you perform a task that needs a larger context size, split the task into smaller steps and run each of them in a new `LanguageModelSession`. For example, to generate a summary for a long article on device:

1. Separate the article into smaller sections.
2. Summarize each section with a new session instance.
3. Combine the sections.
4. Repeat the steps until you get a summary with the size you want, and consider adding the summary to the prompt so it conveys the contextual information.

Use Instruments to analyze token consumption while your app is running and to look for opportunities to improve performance, like with `prewarm(promptPrefix:)`. To profile your app with Instruments:

1. Open your Xcode project and choose Product > Profile to launch Instruments.
2. Select the Blank template, then click Choose.
3. In Instruments, click "+ Instrument" to open the instruments library.
4. Choose the Foundation Models instrument from the list.
5. Choose File > Record Trace. This launches your app and starts a recording session to observe token usage from your app's model interactions.

Because some generation tasks can be resource intensive, consider profiling your app with other instruments — like Activity Monitor and Power Profiler — to identify where your app might be using more system resources than expected.

For more information on managing the context window size, see TN3193: Managing the on-device foundation model's context window.

Topics

Creating a session

`convenience(model:tools:instructions:)`

Start a new session in blank slate state with instructions builder.

`class SystemLanguageModel`

An on-device large language model capable of text generation tasks.

`protocol Tool`

A tool that a model can call to gather information at runtime or perform side effects.

`struct Instructions`

Details you provide that define the model's intended behavior on prompts.

Creating a session from a transcript

`convenience init(model: SystemLanguageModel, tools: [any Tool], transcript: Transcript)`

Start a session by rehydrating from a transcript.

`struct Transcript`

A linear history of entries that reflect an interaction with a session.

Preloading the model

`func prewarm(promptPrefix: Prompt?)`

Loads the resources required for this session into memory, and optionally caches a prefix of your prompt to reduce request latency.

Inspecting session properties

`var isResponding: Bool`

A Boolean value that indicates a response is being generated.

`var transcript: Transcript`

A full history of interactions, including user inputs and model responses.

Generating a request

`func respond(options: GenerationOptions, prompt: () throws -> Prompt) async throws -> LanguageModelSession.Response<String>`

Produces a response to a prompt.

`func respond<Content>(generating: Content.Type, includeSchemaInPrompt: Bool, options: GenerationOptions, prompt: () throws -> Prompt) async throws -> LanguageModelSession.Response<Content>`

Produces a generable object as a response to a prompt.

`func respond(schema: GenerationSchema, includeSchemaInPrompt: Bool, options: GenerationOptions, prompt: () throws -> Prompt) async throws -> LanguageModelSession.Response<GeneratedContent>`

Produces a generated content type as a response to a prompt and schema.

`func respond(to:options:)`

Produces a response to a prompt.

`func respond(to:generating:includeSchemaInPrompt:options:)`

Produces a generable object as a response to a prompt.

`func respond(to:schema:includeSchemaInPrompt:options:)`

Produces a generated content type as a response to a prompt and schema.

`struct Prompt`

A prompt from a person to the model.

`struct Response`

A structure that stores the output of a response call.

`struct GenerationOptions`

Options that control how the model generates its response to a prompt.

Streaming a response

`func streamResponse(to:options:)`

Produces a response stream to a prompt.

`func streamResponse(to:generating:includeSchemaInPrompt:options:)`

Produces a response stream to a prompt and schema.

`func streamResponse(to:schema:includeSchemaInPrompt:options:)`

Produces a response stream to a prompt and schema.

`func streamResponse(options: GenerationOptions, prompt: () throws -> Prompt) rethrows -> sending LanguageModelSession.ResponseStream<String>`

Produces a response stream to a prompt.

```
func streamResponse<Content>(generating: Content.Type, includeSchemaIn
Prompt: Bool, options: GenerationOptions, prompt: () throws -> Prompt)
rethrows -> sending LanguageModelSession.ResponseStream<Content>
```

Produces a response stream for a type.

```
func streamResponse(schema: GenerationSchema, includeSchemaInPrompt:
Bool, options: GenerationOptions, prompt: () throws -> Prompt) rethrows
-> sending LanguageModelSession.ResponseStream<GeneratedContent>
```

Produces a response stream to a prompt and schema.

```
struct ResponseStream
```

An async sequence of snapshots of partially generated content.

```
struct GeneratedContent
```

A type that represents structured, generated content.

```
protocol ConvertibleFromGeneratedContent
```

A type that can be initialized from generated content.

```
protocol ConvertibleToGeneratedContent
```

A type that can be converted to generated content.

Generating feedback

```
func logFeedbackAttachment(sentiment: LanguageModelFeedback.Sentiment?,
issues: [LanguageModelFeedback.Issue], desiredOutput: Transcript.Entry
?) -> Data
```

Logs and serializes data that includes session information that you attach when reporting feedback to Apple.

```
func logFeedbackAttachment(sentiment: LanguageModelFeedback.Sentiment?,
issues: [LanguageModelFeedback.Issue], desiredResponseContent: (any
ConvertibleToGeneratedContent)?) -> Data
```

```
func logFeedbackAttachment(sentiment: LanguageModelFeedback.Sentiment?,
issues: [LanguageModelFeedback.Issue], desiredResponseText: String?) ->
Data
```

Getting the error types

```
enum GenerationError
```

An error that may occur while generating a response.

`struct` `ToolCallError`

An error that occurs while a system language model is calling a tool.

Relationships

Conforms To

`Copyable`
`Observable`
`Sendable`
`SendableMetatype`

See Also

Prompting

`struct` `Instructions`

Details you provide that define the model's intended behavior on prompts.

`struct` `Prompt`

A prompt from a person to the model.

`struct` `Transcript`

A linear history of entries that reflect an interaction with a session.

`struct` `GenerationOptions`

Options that control how the model generates its response to a prompt.