Function

withTaskGroup(of:returning:isolation:body:)

Starts a new scope that can contain a dynamic number of child tasks.

iOS 13.0+ | iPadOS 13.0+ | Mac Catalyst 13.0+ | macOS 10.15+ | tvOS 13.0+ | visionOS 1.0+ | watchOS 6.0+

```
@backDeployed(before: macOS 15.0, iOS 18.0, watchOS 11.0, tvOS 18.0, visionOS 2.0)
func withTaskGroup<ChildTaskResult, GroupResult>(
    of childTaskResultType: ChildTaskResult.Type = ChildTaskResult.self,
    returning returnType: GroupResult.Type = GroupResult.self,
    isolation: isolated (any Actor)? = #isolation,
    body: (inout TaskGroup<ChildTaskResult>) async -> GroupResult
) async -> GroupResult where ChildTaskResult : Sendable
```

Discussion

A group *always* waits for all of its child tasks to complete before it returns. Even canceled tasks must run until completion before this function returns. Canceled child tasks cooperatively react to cancellation and attempt to return as early as possible. After this function returns, the task group is always empty.

To collect the results of the group's child tasks, you can use a `for-await-in` loop:

```
var sum = 0
for await result in group {
    sum += result
}
```

If you need more control or only a few results, you can call `next()` directly:

```
guard let first = await group.next() else {
    group.cancelAll()
    return 0
}
let second = await group.next() ?? 0
group.cancelAll()
return first + second
```

Refer to TaskGroup documentation for detailed discussion of semantics shared between all task groups.

See Also

TaskGroup

See Also

Tasks

`struct Task`

A unit of asynchronous work.

`struct TaskGroup`

A group that contains dynamically created child tasks.

`struct ThrowingTaskGroup`

A group that contains throwing, dynamically created child tasks.

`func withThrowingTaskGroup<ChildTaskResult, GroupResult>(of: ChildTaskResult.Type, returning: GroupResult.Type, isolation: isolated (any Actor)?, body: (inout ThrowingTaskGroup<ChildTaskResult, any Error>) async throws -> GroupResult) async rethrows -> GroupResult`

Starts a new scope that can contain a dynamic number of throwing child tasks.

`struct TaskPriority`

The priority of a task.

`struct DiscardingTaskGroup`

A discarding group that contains dynamically created child tasks.

```
func withDiscardingTaskGroup<GroupResult>(returning: GroupResult.Type,
isolation: isolated (any Actor)?, body: (inout DiscardingTaskGroup)
async -> GroupResult) async -> GroupResult
```

Starts a new scope that can contain a dynamic number of child tasks.

```
struct ThrowingDiscardingTaskGroup
```

A throwing discarding group that contains dynamically created child tasks.

```
func withThrowingDiscardingTaskGroup<GroupResult>(returning: Group
Result.Type, isolation: isolated (any Actor)?, body: (inout Throwing
DiscardingTaskGroup<any Error>) async throws -> GroupResult) async
throws -> GroupResult
```

Starts a new scope that can contain a dynamic number of child tasks.

```
struct UnsafeCurrentTask
```

An unsafe reference to the current task.