orphan:

Unsupported Optimization Attributes

The following attributes are experimental optimizer directives. The attribute names are underscored because they are internal compiler directives. These are not supported language features, have not gone through the Swift Evolution process, and future versions of the compiler are likely to drop support, requiring manual intervention of the source maintainer.

@inline(__always)

Force inlining (at least at -O for now). See the discussion in docs/TransparentAttr.md.

@_specialize directs the compiler to specialize generics

The compiler only automatically specializes generic code if the call site and the callee function are located in the same module. However, the programmer can provide hints to the compiler in the form of @_specialize attributes. For details see :ref:`generics-specialization`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\swift-main\docs\proposals\[swift-main][docs][proposals]UnsupportedOptimizationAttributes.rst`, **line 24**); ***backlink***

Unknown interpreted text role "ref".

This attribute instructs the compiler to specialize on the specified concrete type list. The compiler inserts type checks and dispatches from the generic function to the specialized variant. In the following example, injecting the @_specialize attribute speeds up the code by about 10 times.

```
/// ---------------
/// Framework.swift

public protocol Pingable { func ping() -> Self }
public protocol Playable { func play() }

extension Int : Pingable {
  public func ping() -> Int { return self + 1 }
}

public class Game<T : Pingable> : Playable {
  var t : T

  public init (_ v : T) {t = v}

  @_specialize(where T == Int)
  public func play() {
    for _ in 0...100_000_000 { t = t.ping() }
  }
}

/// -----------------
/// Application.swift

Game(10).play()
```