```
package dsl.html.cont


@DslMarker
annotation class HtmlTagMarker

interface Element

sealed class Tag(val name: String): Element {
   protected val children = arrayListOf<Element>()

   protected fun <T : Element> initTag(tag: T, init: T.() -> Unit): T {
       tag.init()
       children.add(tag)
       return tag
   }

   override fun toString(): String =
       children.joinToString(
           prefix = "<$name>\n",
           postfix = "</$name>\n",
           separator = "",
           transform = Any::toString
       )
}

abstract class TagWithText(name: String): Tag(name) {
   operator fun String.unaryPlus() {
       children.add(TextElement(this))
   }
}

class TextElement(val text: String): Element {
   override fun toString() = "$text\n"
}
```

```
@HtmlTagMarker
class HTML : Tag("html") {
   fun head(init: Head.() -> Unit) = initTag(Head(), init)
   fun body(init: Body.() -> Unit) = initTag(Body(), init)
}

@HtmlTagMarker
class Head : Tag("head") {
   fun title(init: Title.() -> Unit) = initTag(Title(), init)
}

@HtmlTagMarker
class Title : TagWithText("title")

@HtmlTagMarker
class Body : Tag("body") {
   fun p(init: Paragraph.() -> Unit) = initTag(Paragraph(), init)
}

@HtmlTagMarker
class Paragraph : TagWithText("p")

fun html(init: HTML.() -> Unit): HTML {
   val html = HTML()
   html.init()
   return html
}

fun main() {
   val markup =
       html {
           head {
               title { +"Document Title" }
           }
           body {
               p { +"Hello, World!" }
           }
       }
   println(markup)
}
```