

Módulos y Librerías

Curso 2003/04, Fecha:28/11/2003

Enunciado 1 (html) *Crear una librería que permita generar páginas HTML. El nivel de detalle de las funciones de la librería es opcional. A continuación se presenta un posible esquema de dicha librería*

```
> module HTML (HTML, writeHTML, makeHTML, a, p) where
> data HTML = HTML String String
> writeHTML :: FilePath → HTML → IO ()
> writeHTML file (HTML head body) =
>   writeFile file ("<html><head>" ++ head ++
>     "</head><body>" ++ body ++
>     "</body></html>")
> makeHTML :: String → String → HTML
> makeHTML head body = HTML head body
> a :: String → String → String
> a url cs = "<a href=" ++ url ++ ">" ++ cs ++ "</a>"
> p :: String → String
> p cs = "<p>" ++ cs ++ "</p>"
```

Un posible cliente de dicho programa podría ser el siguiente módulo que permite generar una página HTML con información de un grupo de una asignatura.

```
> module CreaHTML where
> import HTML
> main =
>   writeHTML "index.html"
>     (makeHTML "<title>PLF</title>"
>       (p (a "http://petra.euitio.uniovi.es/~dni1" "Nombre_Alumno_1") ++
>         p (a "http://petra.euitio.uniovi.es/~dni2" "Nombre_Alumno_2") ++
>         p (a "mundo.wrl" "Mundo_Virtual"))))
```

Guardar ambos programas en 2 ficheros (con el mismo nombre que el módulo), compilarlos y visualizar la página Web generada al ejecutar la función main.

Incluir los datos de cada alumno del grupo junto con la dirección de su página Web y poner la página generada en una dirección determinada.

Enunciado 2 (vrml) Crear una librería que permita generar páginas VRML. El nivel de detalle de las funciones de la librería es opcional.

A continuación se presenta un posible esquema

```
> module VRML (VRML, Point(..), Color(..),
>               writeVRML, makeVRML,
>               putText, putBox, url) where

> data VRML = VRML String
> data Point = Point Float Float Float
> data Color = RGB Float Float Float

> writeVRML :: FilePath → VRML → IO ()
> writeVRML file (VRML v) =
>   writeFile file (cabecera ++ viewPoint (Point 0 0 20) ++ v)

> cabecera :: String
> cabecera = "#VRML_V2.0_utf8\n\n"

> viewPoint :: Point → String
> viewPoint (Point x y z) = "Viewpoint_{_position_}" ++ sh3 (x,y,z) ++ "}\n"

> makeVRML :: String → VRML
> makeVRML s = VRML s

> url :: String → String → String → String
> url url desc fig =
>   "Anchor_{_description_}" ++ desc ++
>   "\_url_[\" ++ url ++
>   "\_]_children_[\" ++ fig ++ "]"

> putText :: Point → Color → String → String
> putText p c = translate p . color c . verText

> putBox :: Point → Color → (Float, Float, Float) → String
> putBox p c = translate p . color c . verBox

> verText t = "geometry_Text_{_string_}" ++ t ++ "\n"

> verBox (sx, sy, sz) =
>   "geometry_Box_{_size_}" ++ sh3 (sx, sy, sz) ++ "\n"

> translate (Point x y z) s =
>   "Transform_{_translation_}" ++ sh3 (x,y,z) ++
>   "\n_children_[\" ++ s ++ "]"
```

```

> color (RGB r g b) s = "Shape_{_\n" ++ s ++
>   "\n_appearance_Appearance_{_material_Material_{_" ++
>   "\n_{_diffuseColor_" ++ sh3 (r,g,b) ++ "_}}\n"
> sh3 (x,y,z) = show x ++ "_" ++ show y ++ "_" ++ show z ++ "_"

```

Un posible cliente de dicho programa podría ser el siguiente módulo:

```

> module CreaVRML where
> import VRML

> main =
>   writeVRML "mundo.wrl"
>     (makeVRML (url "index.html" "Información"
>                 (putText (Point 0 0 0) (RGB 0 0.1 0.1) "Información" ++
>                 putBox (Point 4 (-2) (-0.2)) (RGB 0.8 1 1) (10,10,0.1))))

```

Guardar ambos programas en 2 ficheros (con el mismo nombre que el módulo), compilarlos y visualizar el mundo virtual generado al ejecutar la función main.

Incluir dicho mundo virtual en una dirección Web determinada

Enunciado 3 (SVG(Opcional)) Crear una librería similar a las anteriores para generar ficheros SVG

Enunciado 4 (Octree(Opcional)) Crear otra librería para manipular octrees