```haskell
-- xhi.hs

import Network.CGI
import Text.Html

import Control.Monad
import Control.Monad.IO.Class
import Data.Maybe (fromJust)

-- import System.Time (getClockTime)

-- BEGIN HACK: utf-8 -> &#x; ---------------------------------------------------

import Data.Char (isAscii, ord)

s2m [] = ""
s2m (c:cs) = c2m c ++ s2m cs

c2m c | isAscii c = [c]
      | otherwise = "&#" ++ show (ord c) ++ ";"

enc = primHtml . s2m -- UniCodeStringToHTML

-- END HACK -------------------------------------------------------------------

{-  renderHtml использует show, show экранирует не-ASCII...

    См. перекодировку:
    http://blog.kfish.org/2007/10/survey-haskell-unicode-support.html
    ... the generic show method does not serialize Strings as UTF-8
    (when using GHC) ...

    Show class hack:
    http://stackoverflow.com/questions/5535512/how-to-hack-ghci-or-hugs-so-that-it-prints-unicode-chars-unescaped

    Альтернативы:
        * писать свой show;
        * использовать только putStr etc. (в т.ч. с файлом-шаблоном)
-}

main = runCGI $ handleErrors cgiMain


cgiMain = do
    email <- getInput "email"
    text  <- getInput "text"
    liftIO $ save email text
    content <- liftIO $ readFile "data"
    output $ renderHtml $ thePage content -- ""


save (Just email) (Just text@(_:_)) = -- Non-empty text (the simplest check)
    {- кириллица автоматически получится перекодированной
       благодаря CGI
    -}
    appendFile "data" $ show $ aRecord
    where
        aRecord =
            p << email
            +++
            p << enc text
            +++
            hr
save _ _ = return ()

-- HTML renderer --------------------------------------------------------------

thePage content = header << theHeader +++ body << theBody content

theHeader =
    -- meta ! [HtmlAttr "charset" "utf-8"] -- не исп., т.к. кириллица кодирована
    thetitle << enc "Объявления"

```

```
74  theBody content =
75      h1 << enc "Объявления"
76      +++
77      hr -- <hr>
78      +++
79      primHtml content -- existing announces
80      +++
81      h2 << enc "Написать объявление"
82      +++
83      form ! [ method "post"
84             , action "/cgi-bin/xhi"
85             ] << [ p << [ primHtml "E-mail: "
86                         , input ! [thetype "text", name "email"]
87                         ]
88                  , p << [ enc "Текст объявления:" +++ br
89                         , textarea ! [name "text", cols "40", rows "5"] << ""
90                         ]
91                  , p << [ button "reset"  $ enc "Очистить"
92                         , spaceHtml --  
93                         , button "submit" $ enc "Отправить"
94                         ]
95                  ]
96
97  button t nested =
98      tag "button" nested ! [ thetype t ] -- <button type="...">...</button>
99
100 -- Not used;
101 -- see maybe (not Maybe!) usage in
102 -- Practical_web_programming_in_Haskell#Getting_user_input
103 theErrorPage =
104     header << thetitle <<  enc "Ошибка заполнения формы"
105     +++
106     body << h2 << enc "Ошибка заполнения формы"
```