

(۱) مقدمه

برای سالیان متوالی سیستم پست الکترونیکی عمومی ترین ابزار کاربردی بر روی شبکه اینترنت به شمار می رفت تا آنکه در اوائل دهه نود میلادی تور جهان گستر^۱ متولد شد. گزاره نیست اگر تور جهان گستر (یا اختصاراً وب) را وسیعترین و پررونقترین ابزار روی شبکه اینترنت تلقی کنیم؛ بگونه‌ای که در نظر کاربران معمولی، وب و اینترنت مفهومی معادل دارد. امروزه صحبت از تور جهان گستر (وب)، صفحه^۲ وب^۳، ابرمتن^۴، آدرسهای حوزه و پروتکل انتقال ابرمتن^۵ بر سر هر زبانی شنیده می‌شود و بقدری تب آن همه جا را فراگرفته که گاه مهندسين شبکه نیز مفاهيم اين اصطلاحات را رها کرده و به کاربردهای آن دل بسته‌اند.

در این فصل پس از معرفی ساده و مفهومی هر یک از اصطلاحات فوق به سمت تعریف پروتکل‌های حاکم بر تور جهان گستر پیش خواهیم رفت.

(۱-۱) مفهوم تور جهان گستر

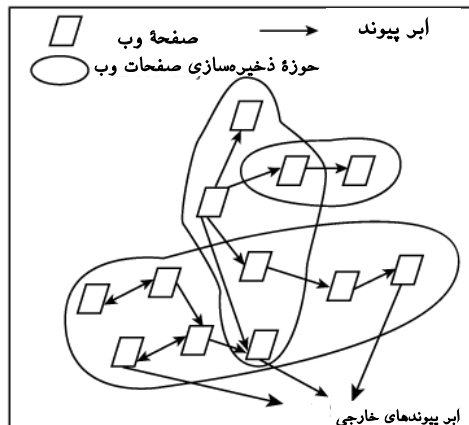
تور جهان گستر یا وب یک روش معماری یا بعبارتی یک نظام برای ذخیره‌سازی و دسترسی به مستندات به هم پیوندخورده‌ای است که روی هزاران هزار ماشین در کل جهان پراکنده و توزیع شده‌اند. هر یک از این مستندات پیوندخورده که شامل متن، صدا و تصاویر گرافیکی و تصاویر متحرک می‌شود، می‌تواند به یک سند دیگر در محلی متفاوت در جهان اشاره نماید. این روش برای دسترسی به اطلاعات توزیع شده در عرض چند سال چنان جاذبه‌ای ایجاد کرد که کل دنیا تحت تاثیر آن قرار گرفت، بگونه‌ای که مردم عادی را نیز واداشت که به این پدیده نوظهور بعنوان ابزاری برای زندگی راحتتر نگاه کنند.

بزرگترین حسن وب، سادگی استفاده از آن است؛ کاربر با وارد کردن آدرس یک سایت که یک صفحه وب در آنجا ذخیره شده، آنرا بر روی کامپیوتر خود منتقل می‌کند، آنرا نگاه و مطالعه کرده و اگر تمایل به دریافت اطلاعات بیشتری در مورد آیتم‌هایی که پررنگ^۵ هستند، داشته باشد با ماوس خود روی آن کلیک می‌کند.

^۱ Web
^۲ Web page
^۳ Hypertext
^۴ Hype Text Transfer Protocol/URL : HTTP
^۵ Highlight

(آیتمهای پر رنگ نقطه پیوند^۱ صفحه فعلی با یک صفحه وب دیگر به حساب می آیند) صفحه جدید هم مثل صفحه قبلی روی کامپیوتر او بار شده و همین روند ادامه می یابد. ممکن است کاربر صفحه ای را از یک سایت در آمریکا بار کرده باشد. پس از مطالعه آن به موضوعی علاقمند می شود که بصورت پر رنگ علامتگذاری شده است؛ روی آن کلیک می کند، بدون آنکه بداند صفحه وب مورد نظر او روی سایتی مثلاً در اروپا قرار دارد. در حقیقت در صفحه اولی که او نگاه می کند در ذیل عنوان پر رنگ یک آدرس دیگر (که از این بعد آنرا URL می نامیم)، وجود داشته که ارتباط بین دو صفحه را برقرار کرده است. در شکل (۱-۱۰) شمای کلی ارتباطات بین صفحات وب به تصویر کشیده شده است.

حال شما تصور کنید اکنون که (ابتدای سال ۱۳۸۰) طبق آمار غیر رسمی حدود یک و نیم میلیارد صفحه وب در کل دنیا ایجاد و ذخیره شده و بین هر یک از آنها دهها پیوند برقرار است، آیا این صفحات و ارتباطات آنها بصورت یک تار عنکبوت که روی کل جهان کشیده شده است مجسم نمی شود؟ اصطلاح وب (تار عنکبوت) از همین تصور نشأت گرفته است. به هر حال ما بدنبال وجه تسمیه و جنبه های ادبی قضیه نیستیم و باید مفاهیم فنی آنرا بررسی نماییم.



شکل (۱-۱۰) شمای کلی ارتباطات بین صفحات وب

^۱ Link

شاید وب جالبترین جنبه استفاده از مفهوم برنامه‌های سرویس‌دهنده / مشتری در شبکه اینترنت باشد؛ بدینگونه که برنامه‌ی سمت مشتری تقاضایی را برای دریافت یک صفحه وب یا یک فایل، به سمت سرویس‌دهنده ارسال می‌کند. برنامه‌ی سرویس‌دهنده در صورت امکان این تقاضا را اجابت کرده و داده‌های لازم را ارسال می‌نماید. بد نیست قبل از بررسی برنامه‌ی سرویس‌دهنده^۱ و برنامه‌ی مشتری (مرورگر) تعریف URL را ارائه‌نمائیم.

۲-۱) مفهوم URL^۲

اشاره کردیم که یک صفحه وب می‌تواند شامل یک پیوند به صفحه وب دیگر در هر نقطه از دنیا باشد. هر پیوند در حقیقت آدرس دقیق یک فایل یا صفحه وب محسوب می‌شود.

با توجه به ناهمگون بودن سیستم‌های عامل و کامپیوترها در دنیا، بعنوان یک نیاز بنیادی باید بتوان فایلها و پرونده‌ها را از لحاظ سبک نامگذاری و محل ذخیره آنها بر روی یک ماشین، هماهنگ و استاندارد کرد. یعنی باید یک روش آدرس دهی برای فایلها انتخاب شود به گونه ای که بتواند به سه سوال زیر برای هر فایل در دنیا پاسخ بدهد:

- نام فایل چیست؟
- محل دقیق ذخیره شده فایل کجاست؟ (یعنی روی چه ماشینی و چه زیر شاخه‌ای قرار دارد؟)
- به چه روشی باید به فایل دسترسی داشت و طبق چه قاعده‌ای می‌توان آن فایل را انتقال داد؟

یک روش آدرس‌دهی استاندارد باید بتواند هر فایل را بطور منحصر به فرد و یکتا در دنیا مشخص کند بنحوی که هیچ ابهامی در آن وجود نداشته باشد. URL روشی برای آدرس دهی منابع و فایلها در دنیاست که به هر سه سوال فوق بدون هیچ ابهامی پاسخ می‌دهد. بنابراین در ذهن خود URL را بعنوان یک آدرس استاندارد یا یک سبک آدرس‌دهی تجسم کنید.

^۱ Web Server

^۲ Uniform Resource Locator

آدرس URL که امروزه شما حتی روی پوسته یک بیسکوئیت آنرا می بینید شامل سه قسمت اساسی است :

- ◆ قسمت پروتکل که به آن قاعده انتقال^۱ هم گفته می شود.
- ◆ نام ماشینی که فایل روی آن قرار دارد. (نام حوزه ماشین یا آدرس IP آن)
- ◆ شاخه و نام فایل

بعنوان مثال به آدرس URL زیر دقت کنید:

<http://www.ibm.com/researches/piiii/paper.htm>

آدرس فوق از سه قسمت تشکیل شده است:

◆ فیلد پروتکل که در این مثال **http** است. این فیلد به برنامه سمت مشتری که "مرورگر"^۲ نام دارد تفهیم می کند که برای بارکردن و انتقال فایل باید از کدام پروتکل و مجموعه فرامین استفاده نماید. درحقیقت این فیلد زبان صحبت کردن مرورگر با سرویس دهنده را تعیین می کند. فیلد پروتکل با علامت //: از بقیه آدرس جدا می شود.

◆ نام حوزه ماشینی که فایل مربوطه روی آن ذخیره شده است. در مثال فوق نام حوزه ماشین www.ibm.com است. نام حوزه بین //: تا اولین / بعدی قرار می گیرد.

◆ نام شاخه و نام فایل: در این قسمت که دقیقاً بعد از نام حوزه شروع می شود و تا انتها ادامه می یابد ، نام شاخه ای که فایل درون آن قرار گرفته و همچنین نام فایل درج می شود. در مثال فوق نام فایل [paper.htm](http://www.ibm.com/researches/piiii/paper.htm) است که بر روی شاخه [/researches/piiii/](http://www.ibm.com/researches/piiii/) قرار دارد.

دقت کنید که بر خلاف سیستم عامل DOS و MS-Windows شاخه ها با علامت \ از هم جدا نمی شوند بلکه مشابه سیستم عامل یونیکس با علامت / از یکدیگر تفکیک می شوند. بنابراین بصورت عمومی آدرس URL بصورت زیر تجزیه می شود:

نام فایل.../.../نام شاخه/ نام حوزه ماشین //: نام پروتکل

دو نکته بسیار مهم در مورد آدرس URL وجود دارد:

^۱ Transfer Protocol / Schema
^۲ Browser

اول آنکه اگر هنگام وارد کردن آدرس نام فایل ذکر نشود، سرویس دهنده بصورت پیش فرض نام فایل مورد نظر را `index.htm` در نظر می گیرد. یعنی دو آدرس زیر دقیقاً معادلند:

`http://www.sony.com/`

`http://www.sony.com/index.htm`

با چنین فرضی آدرسها کوتاه می شوند. معمولاً به اولین صفحه ای که کاربر از یک سایت می بیند و با نام `index.htm` ذخیره شده است، "صفحه خانگی"^۱ گفته می شود. طراح صفحه وب، صفحه اصلی یک سایت را بگونه ای طراحی می کند که برای کاربر نقش یک کاتالوگ فهرست دار را بازی کند؛ نام این فایل را `index.htm` می گذارد تا کاربر مجبور نباشد نام دقیق آنرا در آدرس URL وارد نماید و آدرس کوتاه باشد. پس از انجام مراحل بار شدن صفحه خانگی با نام `index.htm`، کاربر این صفحه را مطالعه کرده روی عناوین دلخواهش کلیک می کند. در بطن هر عنوان پررنگ می تواند یک URL مفصل و بزرگ باشد که اگر چه کاربر آنرا نمی بیند ولی وقتی روی آن کلیک می کند یک صفحه دیگر بار می شود؛ بنابراین کاربر مجبور نیست تمام آدرسهای URL را بداند و فقط دانستن آدرس صفحه خانگی برای دسترسی به تمام مستندات آن سایت کفایت می کند. این آدرسها توسط طراح صفحه وب در ذیل هر عنوان، تنظیم شده و یک پیوند به صفحه دیگر محسوب می شود.

نکته دوم آنست که نام حوزه در آدرس URL می تواند با آدرس IP جایگزین شود؛ یعنی اگر آدرس `203.141.207.14` معادل با نام حوزه `www.sony.com` باشد، دو آدرس URL زیر معادلند:

`http://www.sony.com/products.htm`

`http://203.141.207.14/products.htm`

استفاده از آدرسهای IP بجای نام حوزه مرسوم نیست زیرا به راحتی به خاطر سپرده نمی شود ولی در مجموع امکان پذیر است و چون نیازی به ترجمه نام حوزه وجود ندارد، طبیعتاً اندکی سریعتر است.

توصیه مؤکد آن است که سعی کنید تمامی حروف آدرس URL را حروف کوچک وارد کنید مگر آنکه صریحاً بصورت حروف بزرگ معرفی شده باشند.

^۱ Home Page

دقت کنید که در آدرس‌دهی به سبک URL، به غیر از نام و محل دقیق فایل، روش دریافت فایل ذخیره شده نیز به مرورگر تفهیم می‌شود. بعنوان مثال اگر آدرس زیر را داشته باشید فایلی را بر روی کامپیوتر خودتان مشخص می‌کند:

file://utils/av/readme.htm

برخی از پروتکلها که در آدرس URL استفاده می‌شود در جدول (۲-۱۰) معرفی شده است که در آینده مهمترین آنها را تشریح خواهیم کرد.

نام پروتکل	مورد استفاده	مثال
http	انتقال صفحات ابرمتن	http://www.ibm.com/
ftp	انتقال فایل	ftp://ftp.cs.vu.nl.minx/readm
file	فایلهای محلی	file://user/prog.c
news	گروههای خبری	news://comp.os.minix
gopher	گوفر	gopher://gopher.tc.mn.edu
telnet	تِل نِت	telenet://www.w3.org:80

جدول (۲-۱۰) نام پروتکلهای استاندارد در آدرس URL

۲) مقدمه ای بر سیستم وب

کلاً از دیدگاه مسائل فنی، سیستم وب در دو بخش سازماندهی می‌شود:

- برنامه سمت سرویس دهنده وب و برنامه سمت مشتری وب^۱
- پایگاه اطلاعاتی توزیع شده از صفحات ابرمتن و فایل‌های داده مثل صدا، تصویر و ...

صفحه وب چیزی نیست مگر یک فایل متنی بسیار ساده که با زبان علامت‌گذاری HTML تدوین می‌شوند و در بخشهای آتی روش ایجاد آنرا بوسیله یک ویرایشگر متنی (مثل Notepad) توضیح خواهیم داد.

^۱ Web Server/Web Browser

کاری که "مرورگر" بعنوان یک سرویس گیرنده وب انجام می دهد آنست که تقاضای دریافت یکی از این صفحات یا فایلها را در قالب قراردادی استاندارد (به نام پروتکل HTTP) به سمت سرویس دهنده ارسال می کند. در سمت مقابل سرویس دهنده وب این تقاضا را پردازش کرده و در صورت امکان، فایل مورد نظر را برای مرورگر ارسال می کند. مرورگر پس از دریافت فایل ابرمتنی، آنرا تفسیر کرده و بصورت صفحه آرایسی شده روی خروجی نشان می دهد.

اگر فایل ابرمتنی در جایی به فایل صدا یا تصویر پیوند خورده باشد، آنها نیز توسط مرورگر تقاضا شده و پس از دریافت در جای خود قرار می گیرند. بنابراین سرویس دهنده وب را باید یک برنامه سوکت در نظر گرفت که فرامین مشتریها را دریافت، پردازش و در صورت امکان اجرا می کند. برنامه سمت مشتری نیز برنامه سوکتی است که تقاضاها را در قالب فرامین استاندارد، برای سرویس دهنده وب ارسال می کند؛ در ضمن وظیفه تفسیر و نمایش داده های دریافتی را نیز بر عهده دارد.

در ذهن خود دو مفهوم کاملاً مجزای زیر را از هم تفکیک کنید:

- "پروتکل انتقال صفحات ابرمتن"^۱: این پروتکل زبان یا قراردادی برای صحبت کردن مشتری با سرویس دهنده وب (HTTP) است.
- "زبان نشانه گذاری صفحات ابرمتن"^۲: زبانی برای قالب بندی و صفحه آرائی اطلاعات متنی (HTML) است.

ممکن است خواننده نکته گیر به این قضیه اعتراض کند که سرویس دهنده وب را با سرویس دهنده HTTP همسان گرفتیم در صورتیکه سرویس دهنده وب فراتر از سرویس دهنده HTTP است. برای توجیه قضیه خاطر نشان می کنیم که فعلاً کاربردی ترین بخش از سرویس دهنده وب همان سرویس دهنده HTTP است. معرفی زبان نشانه گذاری صفحات ابرمتنی (HTML) را به بخشهای بعد موكول کرده و فعلاً برنامه های سمت سرویس دهنده و سمت مشتری وب (مرورگر) را مورد بررسی قرار می دهیم تا پیوستگی بحث ما با موضوع برنامه نویسی سوکت از دست نرود.

^۱ Hyper Text Transfer Protocol
^۲ Hyper Text Markup Language

۱-۲) برنامه سمت سرورس‌دهنده وب

در سمت سرورس‌دهنده وب، پروسه‌ای وجود دارد که دائماً به پورت شماره 80 گوش می‌دهد و منتظر تقاضای برقراری ارتباط توسط مشتریان (برنامه‌ی مرورگر) می‌باشد. دقت کنید که این برنامه از سوکتهای نوع استریم استفاده می‌کند و ارتباط از نوع TCP است؛ فرامین و داده‌هایی که بین سرورس‌دهنده و مرورگر مبادله می‌شوند تماماً متنی هستند.

بعد از آنکه ارتباط TCP بین برنامه‌ی سرورس‌دهنده و برنامه‌ی مشتری برقرار شد برنامه‌ی مشتری حق دارد یک تقاضا بفرستد و این تقاضا باید در قالب استاندارد HTTP باشد. سرورس‌دهنده این تقاضا را دریافت و پردازش می‌کند و در صورت امکان این آن را اجرا می‌کند. مراحل بار شدن یک صفحه وب در قالب یک مثال ارائه می‌شود. به روند زیر توجه نمایید:

فرض کنید کاربر، URL زیر را در خط آدرس مرورگر خود (مثلاً در برنامه IE5.0 یا Netscape) وارد کرده و کلید \rightarrow را فشار داده باشد:

<http://www.w3.org/hypertext/www/theproject.html>

مرورگر با تحلیل آدرس URL متوجه می‌شود که باید تقاضای فایلی را طبق پروتکل HTTP به سمت سرورس‌دهنده بفرستد. مرحله‌ی که اتفاق می‌افتد به شرح زیر خواهد بود:

الف) مرورگر آدرس URL را تحلیل کرده و قسمتهای پروتکل، آدرس نام حوزه، شاخه و نام فایل را از آن استخراج می‌کند.

ب) مرورگر تقاضای ترجمه آدرس نام حوزه را به DNS ارسال می‌نماید تا آدرس IP ماشین سرورس‌دهنده به دست آید. دراین مثال مرورگر تقاضای ترجمه نام www.w3.org را به DNS ارسال می‌کند. (معادل فراخوانی تابع `gethostbyname()` در برنامه نویسی سوکت)

ج) DNS در پاسخ، آدرس IP معادل با نام حوزه را بر می‌گرداند. فرض کنید در این مثال DNS آدرس IP را 18.23.0.23 برگردانده است.

د) مرورگر یک ارتباط TCP با آدرس 18.23.0.3 و پورت 80 برقرار می‌کند. (معادل فراخوانی تابع `connect()` در برنامه نویسی سوکت)

ه) پس از برقراری ارتباط، رشته کاراکتری زیر که مجموعاً ۳۵ بایت است به سمت سرورس‌دهنده ارسال می‌شود:

“GET /hypertex/www/theproject.html”

GET یکی از فرامین استاندارد HTTP است که در ادامه توضیح خواهیم داد.

(و) سرویس‌دهنده این رشته را دریافت و پس از پردازش آن، فایل theproject.html را از شاخه /hypertext/www/ استخراج کرده و برای مرورگر ارسال می‌نماید.

(ز) مرورگر فایل را دریافت کرده و پس از خاتمه دریافت ارتباط TCP را قطع می‌کند.

(ح) مرورگر فایل ابرمتنی را تفسیر کرده و آنرا روی خروجی نمایش می‌دهد.

(ط) اگر فایل ابرمتنی در جایی دارای تصویر یا صدا باشد به ازای تک‌تک آنها مراحل الف تا ح را تکرار نموده و آنها را به ترتیب دریافت می‌کند. دقت کنید که درون فایل‌های ابرمتنی داده‌های فایل‌های صدا یا تصویر وجود ندارد بلکه فقط نام و محل قرار گرفتن فایل تصویر یا صدا درون آن درج شده است. (این موضوع بر خلاف سیستم پست الکترونیکی است که داده‌های صدا و تصویر در ادامه متن نامه قرار می‌گیرد.)

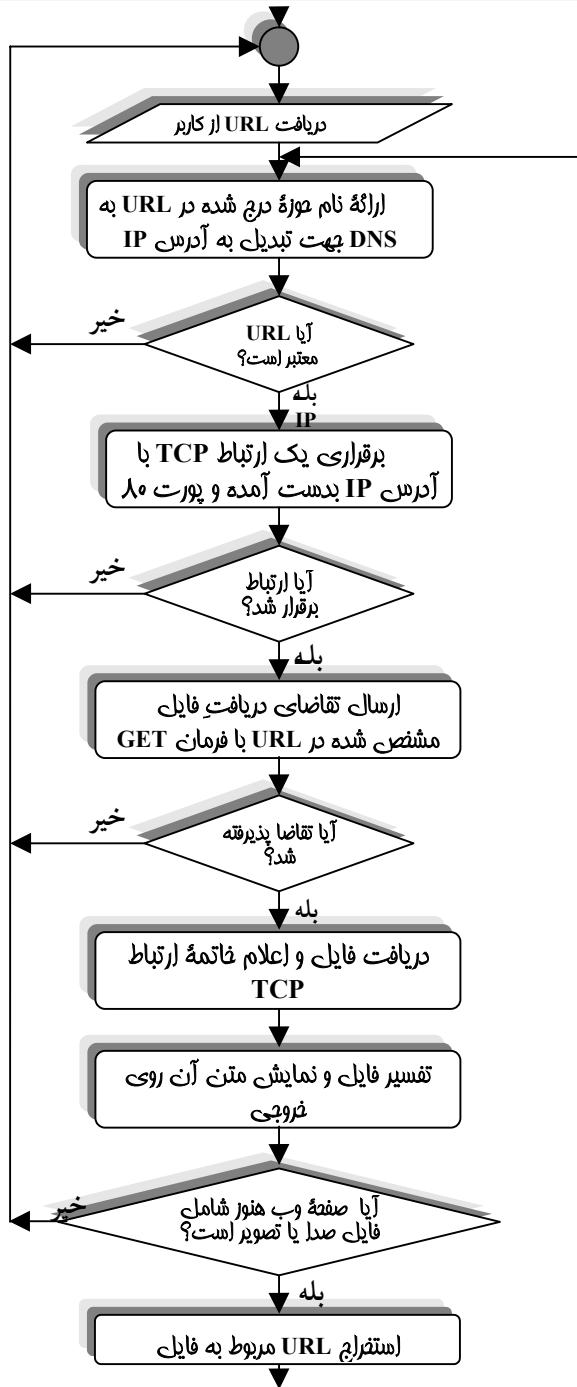
در شکل (۳-۱۰) فلوجارت عملیاتی که مرورگر برای دریافت یک صفحه وب و ملحقات آن انجام می‌دهد، به تصویر کشیده شده است. با تحلیل الگوریتم فوق، روند عملیات برنامه سمت سرویس‌دهنده HTTP مشخص خواهد شد.

سرویس‌دهنده بایستی فرامین دریافتی که بصورت متنی و بر طبق پروتکل HTTP هستند را دریافت، پردازش، احراز هویت و اجرا نماید. در ادامه، پروتکل انتقال صفحات ابرمتن را بررسی می‌نمائیم.

۱۳) پروتکل انتقال ابرمتن : HTTP

پروتکل انتقال ابرمتن مجموعه‌ای از فرامین استاندارد است که از سمت مشتری به سمت سرویس‌دهنده ارسال می‌شود. در حقیقت این پروتکل طریقه صحبت کردن سرویس‌دهنده و مشتری را تبیین کرده است.

بدون مقدمه سراغ مجموعه فرامین در جدول (۴-۱۰) می‌رویم؛ این فرامین در RFC-1945 "متود" نامیده شده است.

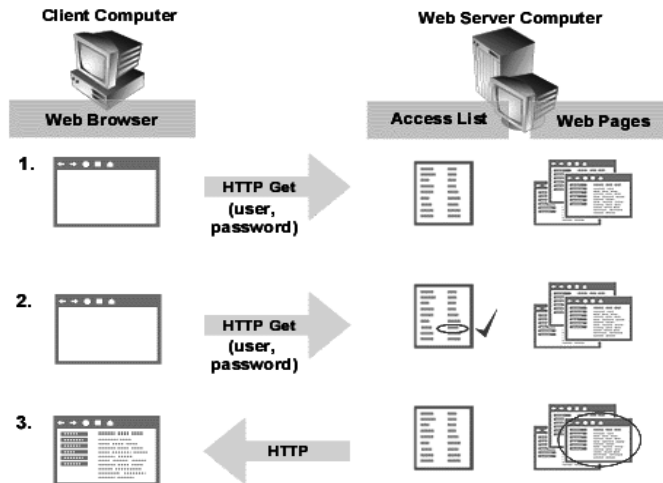


شکل (۳-۱۰) فلوچارت عملیات مرورگر برای دریافت یک صفحه وب

نام فرمان	توضیح
GET	تقاضا برای دریافت یک صفحه وب از سرورس‌دهنده
HEAD	تقاضا برای دریافت سرآیند ^۱ یک صفحه وب
PUT	تقاضا برای ذخیره کردن یک صفحه وب روی یک سرورس‌دهنده
POST	تقاضا برای ضمیمه کردن اطلاعاتی به یک منبع (مثل فایل یا صفحه وب)
DELETE	تقاضا برای حذف یک صفحه وب
LINK	تقاضای برقراری پیوند بین دو منبع موجود
UNLINK	تقاضای خاتمه پیوند دو منبع موجود

جدول (۴-۱۰) فرامین تعریف شده در پروتکل HTTP

◀ **متود GET**: مرورگر با ارسال این متود از سرورس‌دهنده تقاضا می‌کند که یک صفحه وب یا یک فایل دودویی مثل فایل تصویر یا صدا برایش ارسال شود. دقت کنید که فایل‌های صدا یا تصویر در قالب استاندارد MIME ارسال خواهد شد؛ یعنی حتی فایل‌های دودویی نیز بایستی طبق یکی از روشهای تبدیل در استاندارد MIME به حالت متنی تغییر شکل داده شود و سپس ارسال گردد. در جلوی متود GET نام فایل درخواستی درج می‌شود. در شکل (۵-۱۰) محاوره مرورگر و سرورس‌دهنده HTTP برای دریافت یک صفحه وب را نشان می‌دهد.



شکل (۵-۱۰) محاوره مرورگر و سرورس‌دهنده HTTP برای دریافت یک صفحه

اگر بعد از نام فایل درخواستی گزینه "If-Modified-Since:" که در ادامه آن تاریخ و زمان درج شده، اضافه شود سرورس دهنده را وادار می‌کند که فایل درخواستی را به شرطی ارسال نماید که آن فایل بعد از تاریخ و زمان مشخص شده، تغییر کرده باشد. با این گزینه مرورگر می‌تواند در صورتی که قبلاً فایل را دریافت کرده و از آن تاریخ به بعد تغییری نداشته، آنرا از روی ماشین محلی بار کند تا سرعت نمایش صفحات بیشتر شده و اطلاعات بی‌هوده مبادله نشود. مثال:

```
GET /hypertext/www/theproject.html HTTP/1.0 If-Modified- Since: Sat 29 Oct 1999
```

◀ **متود HEAD:** این متود از سرورس دهنده تقاضا می‌کند که فقط سرآیند صفحه وبی را که نام آن در جلوی متود درج شده، ارسال نماید. این متود چند کاربرد دارد: اول آنکه مشخصات صفحه وب، شامل تاریخ آخرین تغییر، عنوان صفحه، نام تدوین کننده و صاحب اصلی آن و برخی از مشخصات اختیاری که در سرآیند صفحه وب درج شده، ارسال می‌شود و این اطلاعات می‌تواند برای مقاصدی همانند تهیه بانک اطلاعاتی از صفحات وب و طراحی "جستجوگرهای وب"^۱ مفید واقع شود.

دوم آنکه می‌توان با این متود صحیح بودن یک URL و وجود یک صفحه وب را ارزیابی کرد.

◀ **متود PUT:** این متود عکس عمل GET است یعنی مرورگر تقاضا می‌کند که یک صفحه وب را بر روی ماشین سرورس دهنده ذخیره نماید. این متود را سرورس دهنده‌هائی حمایت می‌کنند که بخواهند صفحات برخی از کاربران را دریافت کرده ضمن ذخیره، آنها را در اختیار دیگران قرار بدهند. فایل‌هایی که با این متود ارسال می‌شوند باید طبق استاندارد MIME سازماندهی شده باشند.

◀ **متود POST:** این متود از سرورس دهنده تقاضا می‌کند که داده‌هائی را به یک منبع موجود (مثل یک صفحه وب یا یک فایل) اضافه کند. این متود برای ایجاد "صفحات

^۱ Search Engine

آزاد خبری“ ، “تابلو اعلانات^۱“ ، محیطهای نظر خواهی یا ارسال اطلاعات برای یک پروسه تحت وب همانند برنامه های CGI مورد استفاده قرار می گیرد.

◀ **متود DELETE**: این متود از سرویس دهنده تقاضا می کند که یک صفحه وب با نام مشخص را از روی ماشین (ماشین سرویس دهنده) حذف نماید.

دقت شود که بسیاری از سرویس دهنده ها به دلایل امنیتی از متودهای PUT ، POST و DELETE پشتیبانی نمی کنند.

◀ **متودهای LINK و UNLINK**: این دو متود اجازه می دهند که بین دو صفحه وب (یا دو منبع) ارتباط و پیوند برقرار شده یا پیوند قبلی خاتمه داده شود؛. برای توضیح بیشتر در مورد این دو متود به RFC-1945 مراجعه نمایید.

وقتی تقاضا به سمت سرویس دهنده ارسال می شود چه پذیرفته شود و چه پذیرفته نشود ، پاسخی متنی دریافت می دارد که معمولاً بصورت زیر است :

رشته متنی	شماره وضعیت	شماره نسخه/پروتکل
-----------	-------------	-------------------

مثال :

HTTP/1.0 200 OK

یا HTTP/1.0 304 Not Modified

♦ **HTTP/1.0**: نسخه پروتکل را مشخص می کند.

♦ **شماره وضعیت**: شماره ای است سه رقمی که وضعیت اجرای فرمان ارسالی را مشخص می نماید. این شماره سه رقمی بر اساس رقم صدگان به پنج دسته تقسیم می شود:

- 1xx: اطلاعاتی (پاسخی جهت آگاهی بیشتر مشتری)
- 2xx: عمل درخواستی موفقیت آمیز اجرا شده است.
- 3xx: URL مورد تقاضا تغییر آدرس داشته است.
- 4xx: در تقاضای ارسال شده از طرف مشتری خطایی وجود دارد.

^۱ Bulletin Board

• 5xx: در سرویس دهنده خطایی داخلی رخ داده است.
در صورتی که که رقم صدگان ۳، ۴ یا ۵ باشد وضعیت فرمان ارسالی ناموفق بوده است.

♦ رشته متنی: متن کوتاهی که وضعیت اجرای فرمان را به زبان طبیعی توصیف می کند.

برخی از شماره‌های وضعیت در جدول جدول (۶-۱۰) فهرست شده اند.

پس از ارسال اولین خط پاسخ توسط سرویس دهنده در خطوط بعدی نیز یک یا چند سطر اطلاعات اضافی برای مرورگر ارسال می شود که می تواند برای تفسیر داده‌ها بسیار مهم باشد. قالب برخی از این اطلاعات در جدول (۷-۱۰) مشخص شده است. پس از این خطوط که سرآیند نامیده می شوند، یک سطر خالی مرز داده‌های فایل را از سرآیند مشخص کرده و در ادامه داده‌های فایل ارسال می شوند.

شماره	توصیف متنی	توضیح
200	OK	فرمان پذیرفته شد.
204	No content	چیزی وجود ندارد که به مشتری برگشت داده شود.
301	Moved permanently	URL درخواست شده برای همیشه تغییر کرده است.
302	Moved temporarily	URL درخواست شده موقتاً تغییر کرده است.
304	Not modified	عدم تغییر سند درخواستی پس از تاریخ اعلام شده
400	Bad request	فرمان صادره شناخته نشد.
401	Unauthorized	فرمان ارسالی به احراز هویت کاربر نیاز دارد.
403	Not permitted	فرمان صادره غیر مجاز است و اجرا نشد.
404	Not found	صفحه وب یا فایل درخواستی پیدا نشد.
500	Server error	بروز اشکال داخلی در سرویس دهنده
502	Bad gateway	برنامه CGI پاسخ نمیدهد یا وجود ندارد.
503	Service Unavailable	سرویس دهنده به دلیل خاصی فعلاً سرویس نمی دهد.

جدول (۶-۱۰) معرفی مجموعه‌ای از پاسخهای سرویس دهنده HTTP

نوع اطلاعات	قالب اطلاعات
محل دقیق و مطلق منبع یا فایل درخواستی	Location: <i>absoluteURL</i>
نوع و نام سرویس دهنده وب	Server: <i>product</i>
تاریخ و زمان آخرین تغییر منبع یا فایل	Last-Modified: <i>HTTP-date</i>
تاریخ و زمان منقضی شدن منبع یا فایل	Expires: <i>HTTP-date</i>
روش کد گذاری بدنه پیام (استاندارد MIME)	Content-Encoding: <i>encoding</i>
طول داده‌ها بر حسب بایت (استاندارد MIME)	Content-Length: <i>length</i>
نوع سند ارسالی (استاندارد MIME)	Content-Type: <i>type</i>

جدول (۷-۱۰) معرفی مجموعه ای از پاسخهای سرویس دهنده HTTP

بگونه ای که اشاره شد، در پروتکل HTTP همانند پروتکل SMTP، تمام اطلاعات در قالب کاراکترهای آسکی مبادله می‌شوند و هر گاه فایل دودویی (مثل صدا و تصویر) مبادله شود، باید طبق استاندارد MIME (از طریق یکی از دو روش ASCII Armor یا quoted-printable-encoding) به حالت متنی تبدیل شود.

برای آشنائی با چگونگی ارسال داده‌ها از طرف سرویس‌دهنده، در شکل (۸-۱۰) پاسخی را که سرویس‌دهنده HTTP به یک تقاضای دریافت صفحه وب برگردانده، می‌بینیم. این تقاضا بصورت زیر فرستاده شده است:

GET /example/hello.htm HTTP/1.0

```
HTTP/1.0 200 OK
Server: Microsoft-IIS/4.0
Date: Friday, 9-OCT-1998 10:15:00 GMT
Last-Modified: Monday, 5-OCT-1998 21:35:30 GMT
Content-Type: text/html
Content-Length: 98

<html>
<head>
<title>Example of HTTP Response</title>
</head>
<body>
HELLO WORLD!
</body>
</html>
```

شکل (۸-۱۰) ساختار اطلاعات ارسالی از سرویس‌دهنده در پاسخ به تقاضای یک صفحه وب

در شکل (۹-۱۰) مثالی دیگر از چگونگی مبادله فرمان و پاسخ بین برنامه سرویس دهنده HTTP و برنامه مشتری برای دریافت یک صفحه وب نشان داده شده است. بگونه ای که در این شکل مشاهده می کنید ابتدا سعی شده با ماشینی با نام حوزه `www.csc.com` و پورت ۸۰ ارتباط TCP برقرار شود؛ برای اینکار از برنامه Telnet استفاده شده است. برای برقراری ارتباط، برنامه Telnet ابتدا سعی کرده که نام حوزه را به آدرس IP ترجمه کند؛ سپس اقدام به برقراری ارتباط کرده است. در ادامه کاربر توسط متود GET تقاضای یک صفحه وب با نام `toc.html` را صادر کرده است. دقت شود که پس از ارسال رشته فوق در قالب یک خط مجزا، باید یک خط خالی دیگر نیز ارسال شود. (یعنی دو عدد کاراکتر Carriage Return متوالی)

سرویس دهنده پس از دریافت فرمان فوق ابتدا پنج سطر حاوی اطلاعات متنی را بعنوان سرآیند آورده است و سپس با یک خط خالی قسمت سرآیند را خاتمه داده و در ادامه داده های فایل درخواستی ارسال را ارسال کرده است. معنای خطوط سرآیند به شرح ذیل می باشد:

- ◆ **HTTP/1.0 200 Document follows**: تقاضای مرورگر پذیرفته شده و صفحه درخواستی در ادامه ضمیمه می باشد.
- ◆ **MIME-Version: 1.0**: روش سازماندهی اطلاعات مبتنی بر استاندارد MIME نسخه ۱ می باشد.
- ◆ **Server: CERN/3.0**: نام برنامه سرویس دهنده CERN و نسخه آن ۱ است.
- ◆ **Content-Type: text/html**: محتوای داده ها متنی و از نوع HTML می باشد. این گزینه در استاندارد MIME تشریح شد.
- ◆ **Content-Length: 8247**: داده های ارسال شده مجموعاً ۸۲۴۷ بایت می باشد.
- ◆ یک سطر خالی انتهای بخش سرآیند را مشخص می کند.

پس از معرفی "زبان نشانه گذاری صفحات ابرمتن" بازم به سراغ پروتکل HTTP خواهیم آمد.


```

C: telnet www.csc.com 80
T: Trying 18.23.0.23 ...
T: Connected to www.csc.com.
T: Escape character is '^]'.
C: GET /client-server-computing/toc.html HTTP/1.0
C:
S: HTTP/1.0 200 Document follows
S: MIME-Version: 1.0
S: Server: CERN/3.0
S: Content-Type: text/html
S: Content-Length: 8247
S:
S: <html>
S: <head>
S: <meta http-equiv="Content-Type"
S: content="text/html; charset=windows-1256">
S: <meta name="GENERATOR" content="Microsoft FrontPage Express 2.0">
S: <title>Client/Server Computing Table of Contents</title>
S: </head>

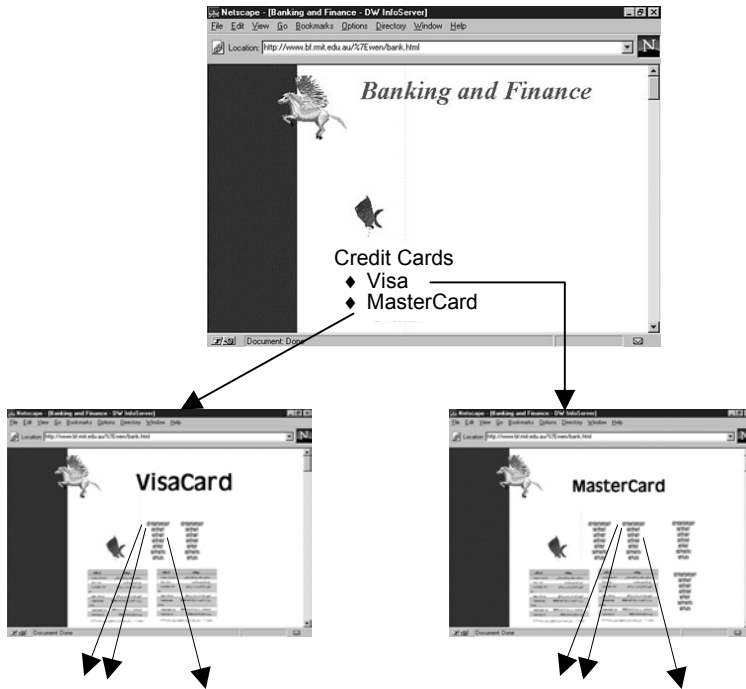
S: <body bgcolor="#FFFFFF" text="#000000" link="#0000FF" vlink="#FF0000">
S:
S: <h1 align="center"><font color="#FF0000">Client/Server Computing </font>
S: <font color="#FF0000" size="4">Second Edition</font></h1>
S:
S: <ul>
S:   <li><ul>
S:     <li><a href="cscfm.htm#18"><font size="3">Foreword</font></a></li>
S:     <li><a href="cscfm.htm#19"><font size="3">Preface</font></a></li>
S:     <li><a href="cscfm.htm#10"><font size="3">Acknowledgments</font></a></li>
S:     <li><a href="cscfm.htm#12"><font size="3">Introduction</font></a></li>
S:   </ul> </li>
S: <li><a href="cs01.htm#1"><font size="3">— 1 —The Business Opportunity</font></a></li>
S: <li><a href="cs02.htm"><font size="3">— 2 —Advantages of Client/Server Computing</font></a></li>
S: <li><a href="cs03.htm"><font size="3">— 3 —Components of Client/Server Applications—The Client</font></a></li>
S: <li><a href="cs04.htm"><font size="3">— 4 —Components of Client/Server Applications—The Server</font></a></li>
S: <li><a href="cs05.htm"><font size="3">— 5 —Components of Client/Server Applications—Connectivity</font></a></li>
S: <li><a href="cs06.htm"><font size="3">— 6 —Client/Server Systems Development—Software</font></a></li>
S: <li><a href="cs07.htm"><font size="3">— 7 —Client/Server Systems Development—Hardware</font></a></li>
S: <li><a href="cs08.htm"><font size="3">— 8 —Client/Server Systems Development—Service and Support</font></a></li>
S: <li><a href="csc09.htm"><font size="3">— 9 —Client/Server Systems Development—Training</font></a></li>
S: <li><a href="csc10.htm"><font size="3">— 10 —The Future of Client/Server Computing</font></a></li>
S: <li><a href="cscxa.htm"><font size="3">— Appendix A —Case Studies</font></a></li>
S: <li><a href="cscxb.htm"><font size="3">— Appendix B —Apple/IBM Joint Venture</font></a></li>
S: <li><a href="cscxc.htm"><font size="3">— Appendix C —Electronic Document Management standards</font></a></li>
S: </ul>
S: </body>
S: </html>

```

شکل (۹-۱۰) محاوره سرویس دهنده و مشتری برای دریافت یک صفحه وب

۱۴) زبان نشانه گذاری ابر متن : HTML

صفحات HTML، متون غنی شده‌ای هستند که اطلاعات موجود در یک سند را بصورت صفحه‌آرایی شده و سازمان‌یافته، در اختیار کاربر قرار می‌دهند. بزرگترین حسن این صفحات آنست که به کاربر این امکان را می‌دهند که به سادگی به صفحه دیگری دسترسی پیدا کند؛ به گونه‌ای که می‌توان مجموعه‌ای از اطلاعات خام را بصورت سلسله‌مراتبی و سطح‌بندی شده در اختیار علاقمندان قرار داد. در شکل (۱۰-۱۰) یک صفحه خانگی بسیار ساده از یک سایت نشان داده شده است. این صفحه دارای دو آیتم است که کاربر با کلیک کردن روی آن، صفحه دیگری را که مرتبط با موضوع دلخواهش می‌باشد، دریافت می‌کند. آن صفحه نیز به صفحات مرتبط دیگری پیوند خورده است؛ این روند ممکن است تا چندین سطح ادامه داشته باشد.



شکل (۱۰-۱۰) یک صفحه خانگی بسیار ساده از یک سایت در اینترنت

HTML (زبان نشانه‌گذاری ابرمتن)، زبانی مانند پاسکال، بیسیک یا C نیست بلکه روشی است که بواسطه آن می‌توان متون خالص و معمولی^۱ را صفحه‌آرایی کرده و عواملی مثل صدا، تصویر، پنجره، منو و فهرستهای انتخاب را به متن اضافه کرد.

HTML حاوی فرامین صفحه‌آرایی است. این فرامین که در بطن متن اصلی قرار می‌گیرد، "برچسب"^۲ نام دارد. برچسبهای درون متن توسط مرورگر تشخیص داده شده و پس از تفسیر، نمایش متون را تحت تاثیر قرار می‌دهند. برچسبهای HTML با علامتهای < > از متن اصلی متمایز می‌شود و عبارتی که در بین آن قرار می‌گیرد، توسط مرورگر از متن اصلی تشخیص داده خواهد شد. تاثیر عملی که با برچسب درون <...> مشخص می‌شود، با علامت </...> لغو می‌گردد. به عنوان مثال متن ساده زیر را در نظر بگیرید:

Internet Engineering

با استفاده از برچسبهای زیر می‌توان مرورگر را وادار کرد که متن را بصورت پررنگ و کج^۳ نشان بدهد:

```
<|><B>Internet Engineering</B></I>
```

مرورگر با تفسیر برچسبها، متن را بصورت زیر نمایش می‌دهد:

Internet Engineering

کسی که با برچسبهای HTML آشنا باشد براحتی می‌تواند با یک ویرایشگر ساده، متن خود را سازماندهی کند. ساختار کلی یک صفحه HTML بصورت زیر است:

```
<HTML>
  <HEAD>
    <TITLE>
      عنوان سند در این ناحیه درج می‌شود.
    </TITLE>
  </HEAD>
  <BODY>
    تمام اطلاعات صفحه وب در این ناحیه درج می‌شود.
  </BODY>
</HTML>
```

^۱ Plain Text

^۲ Tag

^۳ Bold Italic

در ادامه با برخی از برجسته‌های HTML آشنا می‌شویم. هدف از این بخش فقط آشنایی با کلیات HTML است و برای یادگیری اصولی آن باید به مراجع اصلی و کتب تفصیلی مراجعه کرد.

♦ یکی از اصلیتین برجستهها در سازماندهی متون ، برجسته پاراگراف است. مرورگر پاراگراف را بر اساس عرض پنجره نمایش ، تنظیم می‌کند و در صورت لزوم جملات را شکسته و به خطوط بعد می‌برد؛ به این کار پوشش خودکار گفته می‌شود. یک پاراگراف با برجسته <P> آغاز می‌شود و با </P> خاتمه می‌یابد. به مثال زیر دقت کنید:

در اینجا خط بدلیل کوچک بودن عرض صفحه شکسته شده است.

در اینجا خط بدلیل بزرگ بودن عرض صفحه ، شکسته نشده است.

♦ برای ارائه یک مطلب مهم که نظر مخاطب را به خود جلب کند از برجسته استفاده می‌شود؛ در این حالت متنی که تحت تاثیر این برجسته قرار می‌گیرد بصورت کج نشان داده می‌شود. برای تاکید بیشتر می‌توان از برجسته استفاده کرد تا متن را بصورت پررنگ نشان بدهد.

♦ برای آنکه یک سطر را قطع کرده و سطر بعدی از سر خط جدید آغاز شود از برجسته
 استفاده می‌شود. این برجسته می‌تواند عملیات نمایش متن را در سطر جدید بصورت زیر کنترل کند:

<BR CLEAR=LEFT> : خاتمه سطر فعلی و شروع سطر بعدی از سمت چپ به راست

<BR CLEAR=RIGHT> : خاتمه سطر فعلی و شروع سطر بعدی از سمت راست به چپ

<BR CLEAR=ALL> : خاتمه سطر فعلی و شروع سطر بعدی در کل عرض پنجره نمایش

♦ ابرپیوند^۱ : قبلاً اشاره شد که در یک صفحه وب هر آیتیم از متن می‌تواند به صفحه دیگر اشاره کند. برای آنکه قطعه‌ای از متن بصورت "ابریوند" عمل کرده و کاربر با کلیک کردن روی آن صفحه دیگری را دریافت نماید ، از برجسته استفاده می‌شود. به مثال زیر دقت کنید :

 Introduction

^۱ Hyperlink / Hyper Reference

این عبارت باعث می‌شود که کلمه Introduction در متن بصورت پررنگ نشان داده شده و کاربر با کلیک روی آن، مرورگر را وادار کند تا فایل `introduc.html` را بارگذاری^۱ نموده و نمایش بدهد. در زیر یک ابرپیوند کامل را می‌بینید.

```
<A HREF="http://www.w3.org/hypertext/DataSource/www/Geo.html"> Geographical.html
```

♦ برای نمایش تصاویر گرافیکی در متن از برچسب `` استفاده می‌شود. تصاویر فشرده شده نوع GIF و JPEG قابل بارگذاری در یک صفحه وب می‌باشد:

```
<IMG ALIGN = TOP SRC="filename.gif">
```

با این عبارت مرورگر تصویر مشخص شده را بر بالای خط فعلی متن قرار می‌دهد.

```
<IMG ALIGN = MIDDLE SRC="filename.gif">
```

```
<IMG ALIGN = BOTTOM SRC="filename.gif">
```

♦ نمایش فهرستها: در HTML سه نوع فهرست تعریف می‌شود:

• **فهرست بی‌شماره:** این نوع فهرست با برچسب `` و `` مشخص می‌شود و عناوین فهرست، با برچسب `` و `` تفکیک می‌شود. بهترین توضیح مثال زیر است:

```
<UL>
<LI> عنوان ۱ </LI>
<LI> عنوان ۲ </LI>
<LI> عنوان ۳ </LI>
</UL>
```

نمایش این قسمت از متن بصورت زیر است:

- عنوان ۱
- عنوان ۲
- عنوان ۳

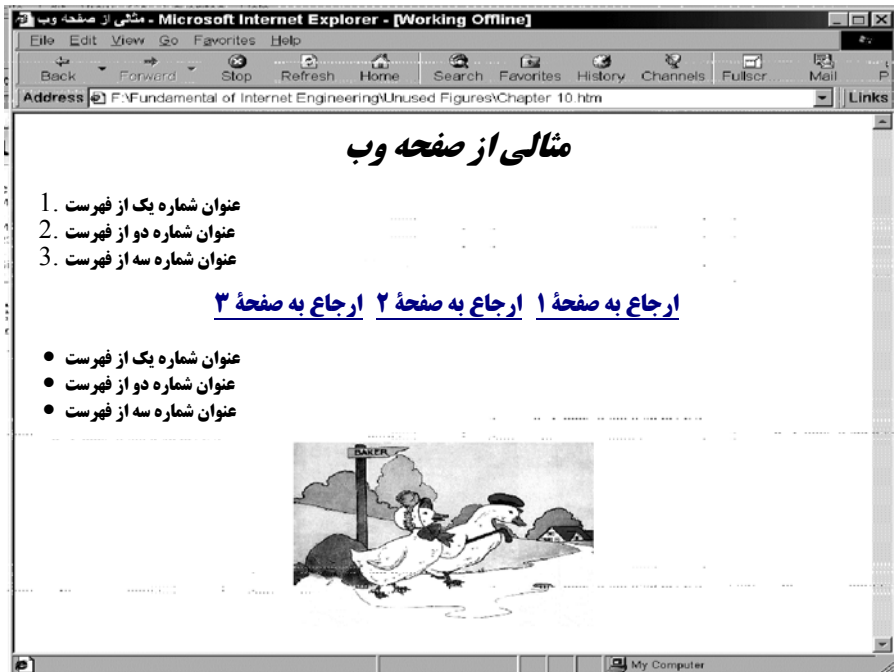
فهرست شماره‌دار: این نوع فهرست مشابه با فهرست قبلی است، با این تفاوت که عناوین فهرست به ترتیب شماره‌گذاری می‌شود. ابتدای فهرست شماره‌دار با `` مشخص می‌شود و برچسب `` برای جداسازی عناوین از یکدیگر می‌باشد؛ پایان فهرست نیز با برچسب `` مشخص می‌گردد. به مثال زیر دقت کنید:

```
<OL>
<LI> عنوان ۱ </LI>
<LI> عنوان ۲ </LI>
<LI> عنوان ۳ </LI>
</OL>
```

نمایش این قسمت از متن بصورت زیر است :

1. عنوان ۱
2. عنوان ۲
3. عنوان ۳

♦ برای تعیین نوع و اندازه قلم (فونت) از برچسب `` استفاده می شود که xxx اندازه قلم و Font Name نام قلم مورد نظر را تعیین می کند. برای آشنایی با برچسبهای فوق به شکل (۱۰-۱۱) دقت کنید. این شکل یک صفحه HTML را در محیط مرورگر نشان می دهد؛ اصل فایل HTML آن در جدول (۱۰-۱۲) نشان داده شده است.



شکل (۱۰-۱۱) یک صفحه HTML در محیط مرورگر

```

<html>
<head>
<title>اصول مهندسی، اینترنت - مثال، از صفحه وب</title>
</head>

<p align="center"><font size="6" face="Nazanin">
<em><strong>وب</strong> از صفحه وب</em></font></p>

<ol>
<li>عنوان شماره یک از فهرست</li>
<li>عنوان شماره دو از فهرست</li>
<li>عنوان شماره سه از فهرست</li>
</ol>
</font></a><font size="5" face="Zar">
<p align="center">
<a href="http://www.malekian.com/page1.html"><strong>۱</strong> ارجاع به صفحه
</strong>
<a href="http://www.malekian.com/page1.html"><strong>۲</strong> ارجاع به صفحه
</strong>
<a href="http://www.malekian.com/page1.html"><strong>۳</strong> ارجاع به صفحه
</strong>

<ul>
<li>عنوان شماره یک از فهرست</li>
<li>عنوان شماره دو از فهرست</li>
<li>عنوان شماره سه از فهرست</li>
</ul>

<p align="center">

</font></p>
</body>
</html>

```

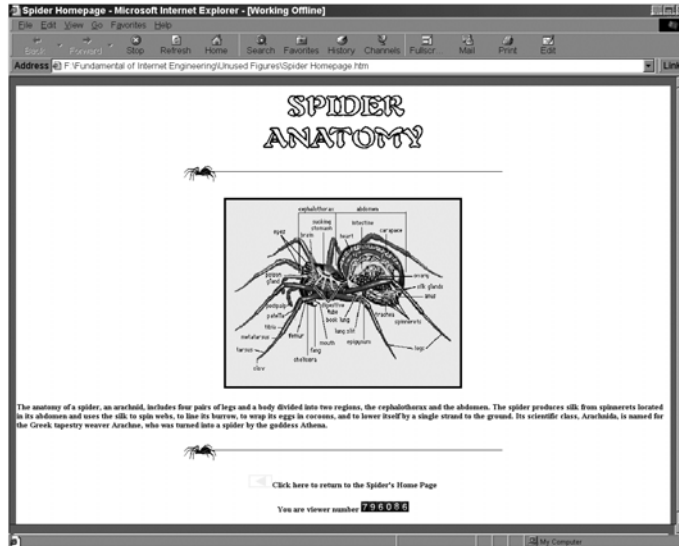
شکل (۱۰-۱۲) فایل HTML مربوط به صفحه وب در شکل (۱۰-۱۱)

۱۴-۱) فرمهای ورود اطلاعات در HTML

آن دسته از صفحات وب که فقط ارائه کننده اطلاعات به کاربر هستند و هیچ کنش و واکنشی با کاربر ندارند، اصطلاحاً "صفحات ایستا"^۱ نامیده می‌شوند. کاربر با دریافت چنین صفحاتی، در زمینه موضوع دلخواه خود اطلاعاتی را از سرویس دهنده دریافت کرده و مطالعه

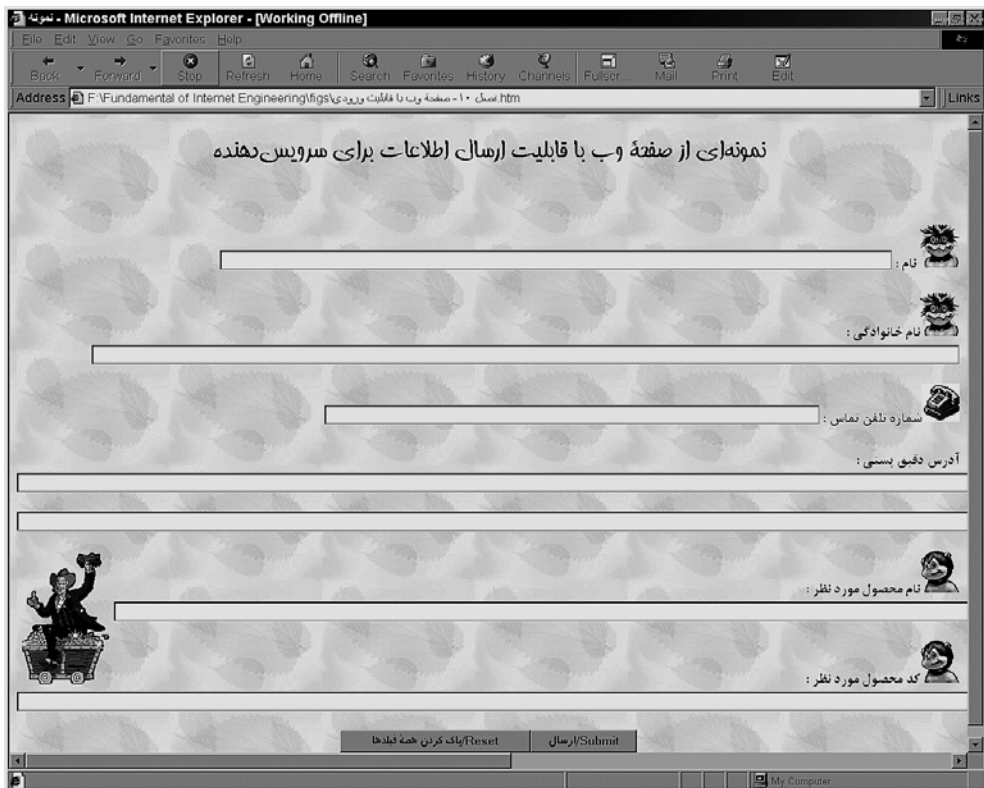
^۱ Static Pages

می‌کند. در شکل (۱۰-۱۳) یک "صفحه وب ایستا" دیده می‌شود. تمام اجزای صفحه به منظور مطالعه کاربر، آراسته شده است.



شکل (۱۰-۱۳) یک "صفحه وب ایستا"

یکی از بزرگترین مشخصه‌های وب آنست که کاربر می‌تواند با یک صفحه وب در تراکش باشد؛ بدین معنا که کاربر قادر است در صفحه وب اطلاعاتی را وارد نموده و آنرا به سمت سرویس‌دهنده ارسال کند. سرویس‌دهنده پس از دریافت اطلاعات و پردازش آن، مجدداً در پاسخ، اطلاعاتی را برمی‌گرداند. برای چنین کاربردهایی صفحه وب باید شامل اجزایی جهت ورود اطلاعات باشد. به عنوان مثال فرض کنید یک شرکت مایل است در سایت وب خود، ضمن معرفی محصولات شرکت، از مشتریان سفارش بگیرد. بنابراین طراح صفحه وب باید نواحی ورود اطلاعات مشتری را در صفحه وب تعریف کند. پس از آنکه کاربر اطلاعات خود را در این نواحی وارد کرد، مرورگر را وادار می‌کند تا آنها را برای سرویس‌دهنده بفرستد. در شکل (۱۰-۱۴) یک صفحه وب با قابلیت ورود و ارسال اطلاعات نشان داده شده است. در جدول (۱۰-۱۵) اصل فایل HTML آن ارائه شده است.



(۱۴-۱۰) یک صفحه وب با قابلیت ورود و ارسال اطلاعات

```

<html>
<head>
<title>نمونه</title>
</head>
<body background=".../SampleWebSite/Nabkgnd.jpg" bgcolor="#FFFFFF">
<p align="center"><font size="6" face="Tabassom"><strong>نمونه‌ای
از صفحه وب با قابلیت ارسال اطلاعات برای سرویس دهنده
</strong></font></p>
<form method="POST" ACTION="http://www.malekian.com/cgi-bin/proc1.pl">
<p align="right"><font face="Nazanin">
<input type="text" size="108" name="Name"><strong> نام :</strong>

<p dir="rtl">

<strong> نام خانوادگی :</strong>
<input type="text" size="140" name="Family">
<p dir="rtl"><strong>

```

ادامه صفحه بعد

```

<strong>شماره تلفن تماس:</strong>
<input type="text" size="79" name="TelNo">
</p>
<p dir="rtl"><strong>آدرس دقیق بستم:</strong>
<input type="text" size="251" name="Addr1">
</p>
<input type="text" size="294" name="Addr2">
</p>
<strong>
<strong>نام محصول مورد نظر:</strong>
<input type="text" size="165" name="ProductNName">
</p>
<strong>
<strong>کد محصول مورد نظر:</strong>
<input type="text" size="170" name="ProductNo">
</p>
<p align="center">
<input type="submit" name="B1" value="Submit/ارسال">
</p> <p>
<input type="reset" name="B2" value="Reset/کردن همه فیلدها">
</p>
</form>
</body>
</html>

```

(۱۰-۱۵) اصل فایل HTML مربوط به صفحه وب شکل (۱۴-۱۰)

در جدول (۱۰-۱۵) تمام برجسبهایی که در زمینه خاکستری نشان داده شده‌اند، مربوط به دریافت اطلاعات از کاربر هستند. این برجسبها را با در نظر گرفتن مثال فوق بررسی می‌کنیم:

♦ برای تعیین ناحیه ورود اطلاعات در یک صفحه وب، از برجسب <FORM ...> استفاده می‌شود. ناحیه ورود اطلاعات با برجسب </FORM> خاتمه می‌یابد. هر صفحه وب می‌تواند چند ناحیه ورود اطلاعات داشته باشد؛ اطلاعات هر ناحیه بطور یکجا برای سرویس‌دهنده ارسال خواهد شد. در هر ناحیه حداقل یک گزینه "تسلیم/ارسال"^۱ وجود دارد که کاربر پس از آنکه اطلاعات این ناحیه را تکمیل کرد، توسط آن به مرورگر فرمان می‌دهد تا آنها را برای سرویس‌دهنده ارسال کند. اطلاعات ارسالی در قالب مشخص و استاندارد تحویل یک پروسه خاص بر روی ماشین سرویس‌دهنده خواهد شد. نام و آدرس این پروسه درون برجسب <FORM ...> مشخص شده است. این پروسه که اصطلاحاً CGI نامیده می‌شود، توسط سرویس‌دهنده فراخوانی شده و اطلاعات ارسالی را دریافت می‌دارد. در ضمن این پروسه

^۱ Submit

وظیفه دارد اطلاعات دریافتی را پردازش و در صورت لزوم ذخیره کند و پاسخهای مناسب را به کاربر برگرداند. (در ادامه CGI را بررسی خواهیم کرد).

برچسب <FORM ...> دارای ویژگیهای زیر است:

- ACTION: آدرس URL مربوط به محل قرار گرفتن برنامه CGI که اطلاعات ارسالی را دریافت و پردازش خواهد کرد.
- METHOD: یکی از متودهای HTTP که توسط آن اطلاعات به سمت سرور ارسال می‌شود. این متودها می‌تواند GET یا POST باشد. [به جدول (۴-۱۰) مراجعه کنید؛ تفاوت این متودها در بخشهای آتی تشریح خواهد شد.]
- ENCTYPE: نوع کدگذاری داده‌های ارسالی را مشخص می‌کند. (چون ارسال اطلاعات از مرورگر به سرور دهنده از استاندارد MIME تبعیت می‌کند لذا در این قسمت نوع کدگذاری داده‌ها مشخص می‌شود-مثل base64-. اگر این خصوصیت معرفی نشود، اطلاعات ارسالی، متن معمولی فرض خواهد شد.)

◆ برچسب <INPUT ...>

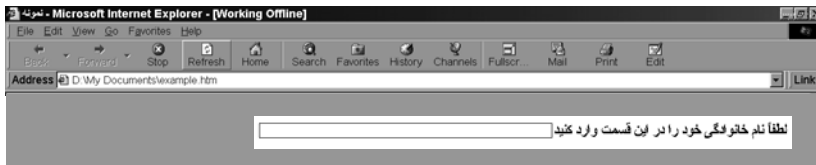
با استفاده از این برچسب می‌توان یکی از عوامل دریافت اطلاعات را در صفحه وب تعریف کرد. ویژگیهای این برچسب عبارتند از:

- TYPE: در صفحه وب می‌توان انواع اشیاء و عوامل ورود اطلاعات را تعریف کرد. با ویژگی TYPE، می‌توان نوع عامل ورودی را تعیین کرد. انواع عوامل ورودی عبارتند از:

TYPE=TEXT: یک فضا برای ورود اطلاعات متنی فراهم می‌آورد. مثال:

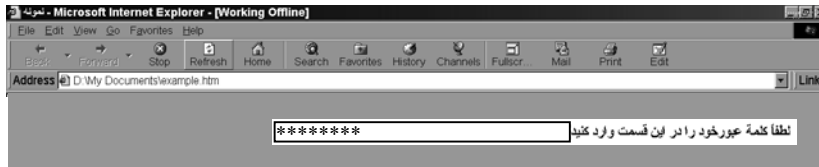
```
<p>
<input type="text" size="152" name="T1">
لطفأ نام خانوادگی خود را در این قسمت وارد کنید<strong>
</p> >
```

نمایش این عامل ورودی در محیط مرورگر به صورت زیر است:



TYPE=PASSWORD : یک فضا برای ورود کلمه عبور فراهم می‌آورد؛ با این تفاوت که در هنگام ورود اطلاعات، محتوای آن دیده نخواهد شد. مثال:

```
<p>
<input type="password" size="152" name="T1">
</p>
<strong>لطفاً کلمه عبور خود را در این قسمت وارد کنید</strong>
```



TYPE=CHECKBOX : یک لیست از گزینه‌ها را تعریف می‌کند تا کاربر بتواند به دلخواه، هر کدام را انتخاب نماید. مثال:

```
<font face="Sina">
<p><input type="checkbox" checked name="C1" value="ON"><strong>گزینه ۱</strong></p>
<p><input type="checkbox" name="C2" ><strong>گزینه ۲</strong></p>
<p><input type="checkbox" name="C3" ><strong>گزینه ۳</strong></p>
<p><input type="checkbox" name="C4" ><strong>گزینه ۴</strong></p>
```



TYPE=RADIO : یک لیست از گزینه‌ها را تعریف می‌کند تا کاربر بتواند به دلخواه، فقط یکی از آنها را انتخاب نماید. مثال:

```
<font face="Sina">
<p><input type="radio" checked name="R1" value="V1"><strong>گزینه ۱</strong></p>
<p><input type="radio" name="R1" value="V2"><strong>گزینه ۲</strong></p>
<p><input type="radio" name="R1" value="V3"><strong>گزینه ۳</strong></p>
<p><input type="radio" name="R1" value="V4"><strong>گزینه ۴</strong></p>
```



TYPE=SUBMIT: یک "کلید فشاری" ^۱ تعریف می کند تا کاربر به مرورگر فرمان بدهد که اطلاعات را به سمت سرور ارسال کند. مثال:

```
<p align="center">
<input type="submit" name="B1" value="ارسال/Submit">
</p>
```

TYPE=RESET: یک "کلید فشاری" تعریف می کند تا کاربر به مرورگر فرمان بدهد که اطلاعات درون تمام عوامل ورودی اطلاعات را پاک نماید. مثال:

```
<p>
<input type="reset" name="B2" value="پاک کردن همهٔ فیلدها/Reset">
</p>
```

در مثال زیر نقش کلید RESET مشخص شده است.



RESET

انتخاب نام برای تمام ورودیها بجز کلیدهای SUBMIT و RESET الزامی است زیرا وقتی مرورگر اطلاعات را برای سرور ارسال می نماید ، نام فیلد مربوطه را هم به همراه

^۱ Push Button

مقدار آن ، ارسال خواهد کرد. برنامه CGI ، وظیفه دارد نام فیلدها و مقادیر آنها را از هم تفکیک کند. در تمام اشیاء و عوامل ورودی ، گزینه VALUE مقدار پیش فرض آنرا تعیین می کند. برای کلیدهای فشاری SUBMIT و RESET ، مقدار گزینه VALUE عنوانی است که بر روی کلیدها ظاهر خواهد شد.

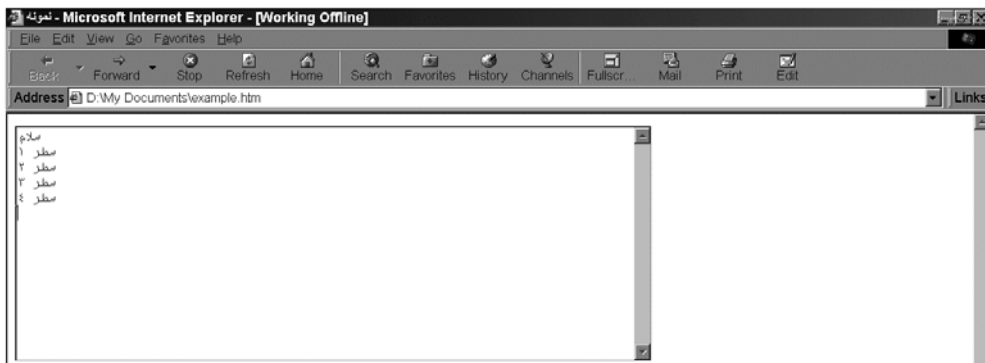
◆ برچسب <TEXTAREA ...>

با استفاده از این برچسب می توان یک فضای چندخطی ورودی در صفحه وب تعریف کرد. این فضا به کاربر اجازه خواهد داد تا متون طولانی خود را وارد کرده ، ارسال نماید. چنین فیلدی می تواند برای کاربردهایی نظیر ورود متن نامه یا نظرسنجی استفاده شود. برخی از ویژگیهای این برچسب عبارتند از:

- NAME : نام فیلد
- ROWS : تعداد سطرهای ورودی
- COLS : تعداد ستونهای ورودی
- DISABLED : ورودی را غیر فعال می کند.

مثال :

```
<p><textarea name="text1" rows="15" cols="80">
</textarea> </p>
```



۱۴-۷) قابلیت‌های دیگر HTML

استاندارد HTML بطور مداوم رو به توسعه و پیشرفت است. پس از معرفی نسخه سوم HTML در ۱۹۹۵ تقریباً هر سال ایده‌های جدیدی در این زمینه ارائه شده است. در نسخه HTML 3.0 قابلیت‌های زیر عرضه شده‌اند:

♦ امکان رسم جدول: جداول یکی از نیازهای اصلی صفحات وب تلقی می‌شوند در حالی که نسخه‌های قدیمی HTML، این امکان را فراهم نکرده بودند. در نسخه ۳، رسم جدول با برچسب `<TABLE ...>` امکان‌پذیر شد. قالب این برچسب بصورت زیر است:

```
<TABLE WIDTH="?" COLS="?" BORDER="?" FRAME="?" CELSPACING="?"
CELLPADDING="?">
```

WIDTH: عرض جدول

COLS: تعداد ستونها

BORDER: ضخامت حاشیه اطراف جدول بر حسب پیکسل

FRAME: حاشیه قابل رویت اطراف جدول

CELLSPACING: فضای خالی میان خانه‌های جدول

CELLPADDING: فضای خالی بین داده‌های درون جدول

پس از تعریف ساختار جدول سطر و ستونهای آن با برچسبهای زیر تعریف می‌شوند:

برچسبهای `<TR>` و `</TR>`: یک سطر از جدول را تعیین می‌کند.

برچسبهای `<TD>` و `</TD>`: محتوی هر یک از خانه‌های جدول توسط یک برچسب

`<TD>` مشخص می‌شود.

به مثال زیر دقت کنید. در این مثال یک جدول ۳×۳ تعریف و به خانه‌های آن مقدار داده

شده است.

```
<font size="6" face="Nazanin">
```

```
<table border="3" cellpadding="3" cellspacing="3">
```

```
<tr>
```

```
<td><p align="center"><strong>۱ ستون ۱ سطر</strong></td>
```

```
<td><p align="center"><strong>۱ ستون ۲ سطر</strong></td>
```

```
<td><p align="center"><strong>۱ ستون ۳ سطر</strong></td>
```

```
</tr>
```

```
<tr>
```

```
<td><p align="center"><strong>۲ ستون ۱ سطر</strong></td>
```

```
<td><p align="center"><strong>۲ ستون ۲ سطر</strong></td>
```

```
<td><p align="center"><strong>۲ ستون ۳ سطر</strong></td>
```

```
</tr>
```

```

<tr>
  <td align="center"><strong>۳ سطر ۱ ستون</strong></td>
  <td align="center"><strong>۳ سطر ۲ ستون</strong></td>
  <td align="center"><strong>۳ سطر ۳ ستون</strong></td>
</tr>
</table>

```



♦ **جاسازی یک APPLET در صفحه وب :** در یک صفحه وب با استفاده از برجسبهای <APPLET ...> می توان اپلت های جاوا را جاسازی کرد. اپلتها ، قابلیت های حرفه ای شبیه مدیریت نقشه های تصویری ، امکانات چندرسانه ای و بازیهای کامپیوتری را به صفحات وب ارائه می کنند. مثال :

```
<APPLET CODE="FirstApplet" WIDTH=300 HEIGHT=200> </APPLET>
```

CODE : نام فایل اپلت

WIDTH : عرض پنجره ای که اپلت در آن اجرا می شود.

HEIGHT : ارتفاع پنجره ای که اپلت در آن اجرا می شود.

♦ **امکانات فرمول نویسی :** در نسخه های جدید HTML ارائه فرمولهای ریاضی در یک صفحه وب امکان پذیر شده است. این قابلیت معمولاً یکی از نیازهای اولیه سایت های علمی و تحقیقاتی تلقی می شود.

در خاتمه این بخش بار دیگر تاکید می کنیم که هدف از معرفی HTML ، فقط آشنایی با اصول آن بوده است و برای آشنایی تفصیلی با آن باید به مراجع آخر فصل مراجعه کنید.

۳-۱۴) مزایای HTML

HTML اولین زبان نشانه‌گذاری متن نبود و در دههٔ نود (دورهٔ تولد و رشد فراگیر HTML) حتی ایدهٔ جدیدی هم محسوب نمی‌شد. رشد HTML ناشی از تواناییهای آن در برآورده کردن نیازهای شبکهٔ اینترنت، در موارد زیر بوده است:

- ◆ **استقلال^۱**: HTML یک استاندارد مبتنی بر کدهای ASCII است و هیچ وابستگی اجرایی به سخت‌افزار و نرم‌افزار ندارد. یک فایل HTML می‌تواند بین دو ماشین کاملاً متفاوت مبادله شود، بدون آنکه هیچ نگرانی در مورد عدم سازگاری ماشینها وجود داشته باشد. هر ماشین با سخت‌افزار و نرم‌افزار خاص خود، با استفاده از مرورگر سازگار با محیط خود، یک فایل HTML را تفسیر کرده و نمایش خواهد داد. با توجه به تنوع ماشینها و سیستمهای عامل در شبکهٔ اینترنت، این خصوصیت یکی از نیازهای بنیادی محسوب می‌شود.
- ◆ **سرعت و سادگی**: فایل‌های HTML از نظر اندازه و حجم فایل، کوچک هستند و برای پایین آوردن ترافیک شبکه، ابزاری مناسب محسوب می‌شوند؛ در عین حال پیچیدگی خاصی ندارند و به راحتی می‌توان یک فایل HTML ایجاد و استفاده کرد. سادگی این استاندارد باعث شد که در زمان بسیار کوتاهی ابزارهای قدرتمند طراحی صفحات وب پدید آید و تولید صفحات وب در حد یک کار تجربی و بدون نیاز به اطلاعات فنی، برای عموم آسان شود.
- ◆ **امکان دریافت اطلاعات از صفحهٔ وب**: صفحات وب ابزار بسیار ساده و مناسبی برای دریافت اطلاعات از کاربر و ارسال آن به سرویس‌دهنده هستند؛ کاری که اگر نیاز باشد مستقل از وب انجام شود، به تخصص و زمان بسیار زیادی احتیاج دارد. این امکان باعث شده که محیط وب از حالت اطلاع‌رسانی صرف درآمده و به یک ابزار مناسب و سریع جهت امور اقتصادی، اداری، تجاری تبدیل شود.
- ◆ **سازمان‌دهی سلسله‌مراتبی**: صفحات وب با استفاده از مفهوم ابریوند، مستندات را بصورت سلسله‌مراتبی و دسته‌بندی شده به متقاضی عرضه می‌کنند. در این روش دسترسی به اطلاعات، بسیار سریع و ساده خواهد شد.
- ◆ **پشتیبانی همگانی**: سادگی و جذابیت وب باعث شد که تمام توسعه‌دهندگان نرم‌افزار، در سیستمهای خود از آن پشتیبانی کنند. امروزه سیستمی را نمی‌توان یافت که از وب پشتیبانی نکند یا مرورگر نداشته باشد. امروزه اکثر بانکهای اطلاعاتی قادرند در محیط وب نیز بکارگرفته شوند. یعنی کاربر از راه دور و با استفاده از مرورگر و استاندارد HTML با آن تراکنش داشته باشد.

^۱ Platform Independence

۵) برنامه‌های CGI^۱

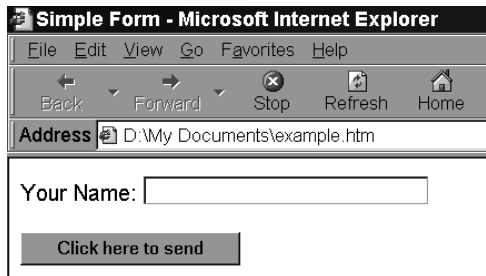
فرض کنید که یک صفحه وب، اطلاعاتی را از کاربر دریافت کند. کاربر می‌تواند با فشار دادن کلید SUBMIT آنها را برای سرویس‌دهنده ارسال نماید. سرویس‌دهنده HTTP فقط وظیفه دریافت یا ارسال داده‌ها را بر عهده دارد و کاری در مورد پردازش آنها انجام نمی‌دهد. حال دو سوال زیر مطرح می‌شود:

♦ کدام برنامه بر روی ماشین سرویس‌دهنده داده‌های ارسالی از مرورگر را دریافت و پردازش می‌نماید؟

♦ مرورگر بر اساس چه الگویی داده‌ها را ارسال می‌کند و برنامه پردازش‌کننده داده‌ها چگونه آنها را از سرویس‌دهنده HTTP تحویل می‌گیرد؟

وقتی طراح صفحه وب، یک ناحیه را برای ورود اطلاعات با برچسب <FORM ...> تعریف می‌کند، موظف است در درون این برچسب آدرس برنامه تحویل‌گیرنده و پردازش‌کننده داده‌ها را دقیقاً مشخص نماید. به عنوان مثال به قطعه کد HTML زیر و نمایش آن در محیط مرورگر دقت کنید.

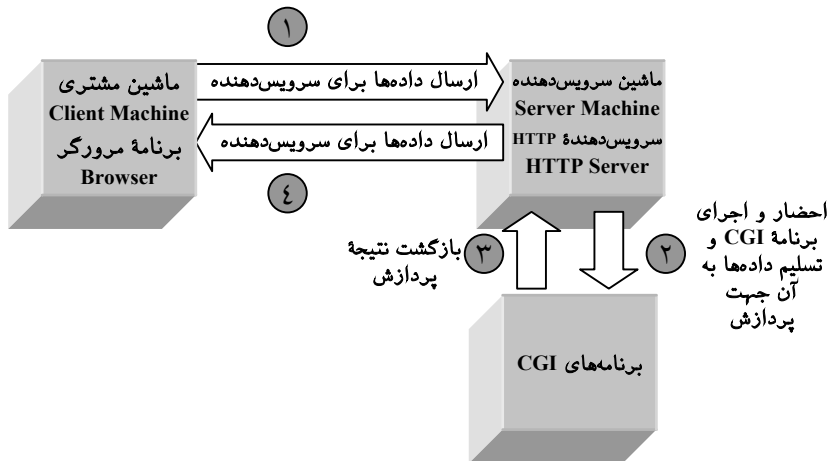
```
<HTML>
<HEAD><TITLE>Simple Form</TITLE></HEAD>
<BODY>
<FORM Method="POST" Action="http://www.abcdef.com/cgi-bin/prog.exe">
Your Name:
<INPUT Name="user" SIZE="30"><P>
<INPUT Type=submit Value="Click here to send">
</FORM>
</BODY>
</HTML>
```



^۱ Common Gateway Interface

در این قطعه کد، نام و محل برنامهٔ تحویل گیرندهٔ داده‌ها با آدرس URL `http://www.abcdef.com/cgi-bin/prog.exe` مشخص شده است. وقتی مرورگر داده‌ها را برای سرویس‌دهندهٔ HTTP ارسال می‌کند، سرویس‌دهنده برنامهٔ `prog.exe` را بارگذاری و اجرا کرده و داده‌ها را تحویل آن می‌دهد. این برنامه قادر است ضمن پردازش داده‌ها، پاسخهای مناسب را در قالب استاندارد HTML تولید کرده و برای کاربر باز پس بفرستد. این برنامه اصطلاحاً CGI نامیده می‌شود. برنامه‌های CGI با نامهای "اسکرپت CGI"^۱ یا "برنامه کاربردی CGI"^۲ هم معرفی شده‌اند.

CGI استاندارد است که چگونگی ارتباط برنامه‌های جانبی با سرویس‌دهندهٔ HTTP را تبیین می‌کند. پروتکل HTTP به تنهایی فقط قادر به ارسال و دریافت صفحات وب است و برنامه‌های CGI در کنار این پروتکل می‌توانند یک ارتباط دوطرفه با کاربر ایجاد نمایند؛ به گونه‌ای که کاربر می‌تواند از راه دور با این برنامه‌ها تراکنش داشته باشد. در حقیقت HTTP به عنوان یک پروتکل واسطه انتقال داده، بین کاربر و این برنامه‌ها انجام وظیفه می‌کند. شکل (۱۶-۱۰) این مفهوم را نشان می‌دهد.



شکل (۱۶-۱۰) تراکنش مرورگر و برنامهٔ CGI از طریق پروتکل HTTP

^۱ CGI Script
^۲ CGI Application

برنامه‌های CGI را می‌توان به زبانهای مختلفی نوشت و نیاز به ابزار خاصی ندارد. زبانهایی که امکان نوشتن برنامه‌های CGI را فراهم آورده‌اند، عبارتند از:

- C ♦
- C++ ♦
- Perl ♦
- Tcl ♦
- Visual Basic (Microsoft Windows) ♦
- Shell Scripts (UNIX) ♦
- Apple Scripts ♦
- Delphi ♦

برنامه‌های CGI، برنامه‌های کاملاً معمولی هستند؛ تنها تفاوت آنها در دریافت داده‌ها از ورودی است. برنامه‌های معمولی داده‌های خود را از طریق صفحه‌کلید یا مؤس دریافت می‌کنند، درحالی که برنامه‌های CGI ورودیهای خود را از سرویس‌دهنده HTTP می‌گیرند. خروجیهای این برنامه‌ها نیز به سرویس‌دهنده HTTP ارسال می‌شود تا برای مشتری فرستاده شده و در محیط مرورگر نشان داده شود. برنامه‌های CGI معمولاً هیچگونه خروجی یا پنجره بر روی ماشین سرویس‌دهنده ندارند. (مگر در موارد خاص)

بطور معمول طراح یک صفحه وب، خودش برنامه CGI متناظر با آن را برنامه‌نویسی می‌کند؛ زیرا نام فیلدها و اشیاء ورودی در یک صفحه وب، باید در برنامه CGI متناظر با آن شناخته شده و تطابق داشته باشد و مرورگر محتوای هر یک از این فیلدها را همراه نام فیلد، ارسال می‌نماید.

در ادامه باید الگوی ارسال داده‌ها را به یک برنامه CGI بررسی کنیم.

۵-۱) الگوهای ارسال اطلاعات برای یک برنامه CGI

به دو روش سرویس‌دهنده HTTP یک برنامه CGI را راه‌اندازی کرده و داده‌های ارسالی از مرورگر را تحویل آن می‌دهد. طراح صفحه وب، در پرچسب <FORM ...>، ضمن معرفی کردن محل و نام برنامه CGI، روش تسلیم داده‌ها را نیز تعریف می‌نماید. این دو روش عبارتند از:

◀ استفاده از الگوی GET :

<FORM Method="GET" Action="http://www.abcdef.com/cgi-bin/prog.exe">

استفاده از الگوی POST :

```
<FORM Method="POST" Action="http://www.abcdef.com/cgi-bin/prog.exe">
```

اگر از الگوی GET برای ارسال داده‌ها استفاده شود، داده‌های جمع‌آوری شده از صفحه وب به آدرس URL آن ضمیمه شده و به سمت سرور ارسال خواهد شد. مثال زیر بسیار گویاست:

```
<font face="Arial">
<form action="http://www.abcdefg.com/cgi-bin/prog.exe" method="GET">
<p>Your Name:
<input type="text" size="30" name="UserName"></p>
<p>Your Last Name <input type="text" size="39" name="UserFamily"></p>
<p><input type="submit" value="Click here to send"> </form></font></p>
```

```
GET /cgi-bin/prog.exe?UserName=Ehsan&UserFamily=Malekian HTTP/1.0
```

به گونه‌ای که نشان داده شده اگر از الگوی GET استفاده شود، مرورگر پس از برقراری ارتباط TCP با سرور دهنده HTTP، از فرمان معمولی GET استفاده می‌کند، با این تفاوت که در ادامه آدرس URL، نام و مقادیر هر فیلد ضمیمه و ارسال می‌شود. وقتی سرور دهنده HTTP این رشته را دریافت می‌کند، برنامه مشخص شده را اجرا کرده و ادامه رشته را (بعد از کاراکتر ؟) در یک "متغیر محیطی"^۱ به نام QUERY_STRING قرار می‌گیرد.^۲ برنامه CGI

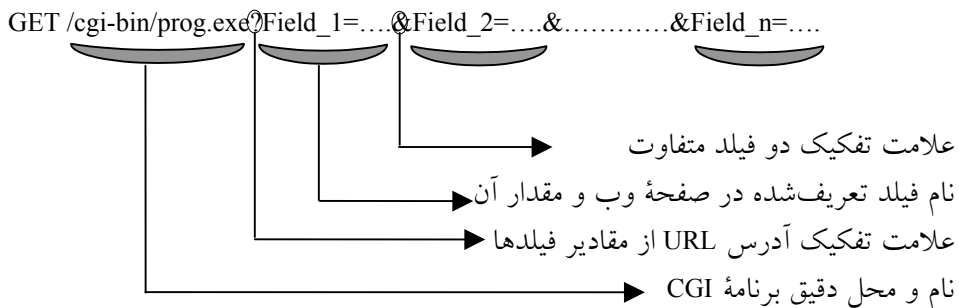
^۱ Environment Variable

^۲ متغیر QUERY_STRING، در برنامه نیاز به تعریف ندارد بلکه با استفاده از تابع getenv(...)، (در زبان C یا توابع معادل در دیگر زبانها) می‌توان به آن دسترسی داشت.

می‌تواند داده‌ها را از این متغیر استخراج، تجزیه و تحلیل^۱ و پردازش نماید. اگر بخواهیم این روش فراخوانی برنامه CGI را با برنامه‌ها معمولی مقایسه کنیم، همانند فراخوانی برنامه در خط فرمان به‌مراه پارامترهای ورودی است؛ در این برنامه‌ها، متغیرهای `argv[]` و `argc` پارامترهای خط فرمان را در اختیار برنامه‌نویس قرار می‌دهند.

الگوی ارسال داده‌های یک صفحه وب بصورت زیر است:

فرض کنید عوامل و اشیاء یک صفحه وب به نامهای فرضی `Field_1` تا `Field_n` نامگذاری شده باشد. هرگاه کاربر کلید `SUBMIT` را فشار داد، رشته زیر تولید شده و ارسال خواهد شد:



رعایت قواعد زیر در تدوین رشته ارسالی، لازم است:

- ◆ محل و نام برنامه CGI با علامت `?` از بقیه رشته جدا می‌شود.
 - ◆ نام هر فیلد و مقدار فیلد با علامت `=` از هم تفکیک می‌شود.
 - ◆ اگر قرار باشد چندین فیلد و مقادیر آن ارسال شود، نام و مقدار هر فیلد با علامت `&` از هم تفکیک می‌شود.
 - ◆ اگر در بین داده‌ها فاصله خالی (Blank) وجود داشته باشد، باید با علامت `+` جایگزین شود.
 - ◆ اگر در بین داده‌ها کاراکترهای ASCII با کد زیر ۳۳ یا یکی از علامتهای `(+ = & % ?)` وجود داشته باشد، ابتدا علامت `%` و سپس کد هگزادسیمال آن بجای آن کاراکتر قرار می‌گیرد. مثلاً کاراکتر با کد `۳۲` بصورت `20%` یا کاراکتر با کد `۳۰` بصورت `1E%` جایگزین می‌شود.
- مثال :

`GET /cgi-bin/prog.exe?Name=Ali+Reza&Family=Ahmadi+Tehrani`

^۱ Parse

استفاده از الگوی GET زمانی مناسب خواهد بود که مجموع کل رشته‌های که به سمت سرور می‌دهند ارسال می‌شود زیر هزار کاراکتر باشد، زیرا سرور می‌دهنده‌های HTTP رشته‌های با طول بیش از هزار کاراکتر را نخواهند پذیرفت.^۱ اکثر برنامه‌های CGI از الگوی POST استفاده می‌کنند.

۵-۲) متغیرهای محیطی قابل استفاده در یک برنامه CGI

به غیر از متغیر محیطی QUERY_STRING، متغیرهای دیگری نیز هستند که برنامه CGI می‌تواند در صورت لزوم از آنها استفاده کند. این متغیرهای محیطی در جدول (۱۷-۱۰) معرفی شده‌اند. برای دسترسی به این متغیرهای در یک زبان برنامه‌نویسی همانند C باید آدرس آنرا با استفاده از تابع سیستمی ("نام متغیر محیطی") getenv بدست آورد. مثال:

```
p = getenv("CONTENT_LENGTH");
if(p != NULL) {...}
else { .... }
```

نام متغیر محیطی	محتوی
REQUEST_METHOD	الگوی ارسال داده‌ها (یکی از دو مقدار GET یا POST)
QUERY_STRING	رشته بعد از علامت ؟ ضمیمه شده به URL (رشته حاوی نام و مقادیر فیلدها)
CONTENT_LENGTH	طول داده‌های ارسالی به برنامه CGI (بر حسب بایت) (اگر الگوی ارسال POST باشد)
CONTENT_TYPE	نوع داده‌های ارسالی به برنامه CGI (مشخصه MIME) (اگر الگوی ارسال POST باشد)
PATH_INFO	مسیر دقیق و محل قرار گرفتن برنامه CGI (مشخص شده در URL)
HTTP_Accept	نوعی از قالب داده‌ها که برنامه مرورگر قادر است، بپذیرد. (مشخصه MIME)
GATEWAY_INTERFACE	نام و نسخه CGI (مثلاً CGI/1.1)
SERVER_PORT	شماره پورت سرور دهنده HTTP که تقاضا به آن ارسال شده است. (بطور معمول ۸۰)
SERVER_PROTOCOL	نام و نسخه سرور دهنده وب (مثلاً HTTP/1.0)
SERVER_SOFTWARE	نام و نسخه نرم‌افزار نصب شده بعنوان سرور دهنده وب (مثلاً IIS/4.0)
SERVER_NAME	نام ماشینی که سرور دهنده وب بر روی آن نصب شده (مثلاً www.ibm.com)
REMOTE_ADDR	آدرس IP ماشین سرور دهنده وب

جدول (۱۷-۱۰) متغیرهای محیطی قابل استفاده در یک برنامه CGI

^۱ در برخی از سرور دهنده‌ها، حداکثر طول یک خط ۲۵۵ کاراکتر و پیش‌فرض آن ۸۰ است.

۳-۵) الگوی POST

برای آن دسته از صفحات وب که حجم نامشخصی از داده‌ها را برای سرویس‌دهنده ارسال می‌کند، برنامه CGI باید از الگوی POST استفاده کند. اگر برای فراخوانی برنامه CGI از الگوی POST استفاده شود، سرویس‌دهنده HTTP، داده‌ها را از طریق متغیر محیطی QUERY_STRING به برنامه نمی‌فرستد بلکه از طریق "ورودی استاندارد" (یعنی همان مفهوم stdin در زبان C) به برنامه هدایت می‌شود. در این حالت برنامه CGI می‌تواند از دستورات معمولی خواندن از کنسول ورودی (مثل تابع fscanf() یا fgetc() در زبان C یا cin در C++) برای دریافت داده‌ها از سرویس‌دهنده اقدام کند.

هر برنامه CGI ممکن است بخواهد برای اطلاع کاربر نتیجه‌ای را به مرورگر برگرداند. واضح است که خروجی یک برنامه CGI باید روی مرورگر کاربر نشان داده شود و اینگونه برنامه‌ها بطور معمول خروجی خاصی بر روی سرویس‌دهنده ندارند. برای برگرداندن اطلاعات به مرورگری که برای یک برنامه CGI داده ارسال کرده است، از "خروجی استاندارد" (یعنی مفهوم stdout در زبان C) استفاده می‌شود. یعنی برای نمایش خروجی‌ها در برنامه CGI می‌توان از دستورات معمولی نوشتن روی کنسول خروجی (مثل تابع printf() در زبان C یا cout در C++) استفاده کرد.

در حقیقت سرویس‌دهنده HTTP هنگامی که یک برنامه CGI را فراخوانی می‌کند، مسیر کنسول ورودی و خروجی استاندارد (stdin و stdout) را به سمت خودش برمی‌گرداند.^۱

در هنگام نوشتن بر روی کنسول خروجی (یعنی زمانی که پیغامی جهت نمایش برای مرورگر ارسال می‌شود) باید به دو نکته اساسی زیر دقت شود:

◀ با توجه به آنکه هر چیزی که با دستورات معمولی مثل printf() بر روی خروجی نوشته می‌شود، بر روی مرورگر نشان داده خواهد شد، لذا پیغامها باید در قالب یک فایل HTML نوشته شوند.

◀ هر خط ارسالی برای مرورگر باید با دو کاراکتر \n\n خاتمه یابد.

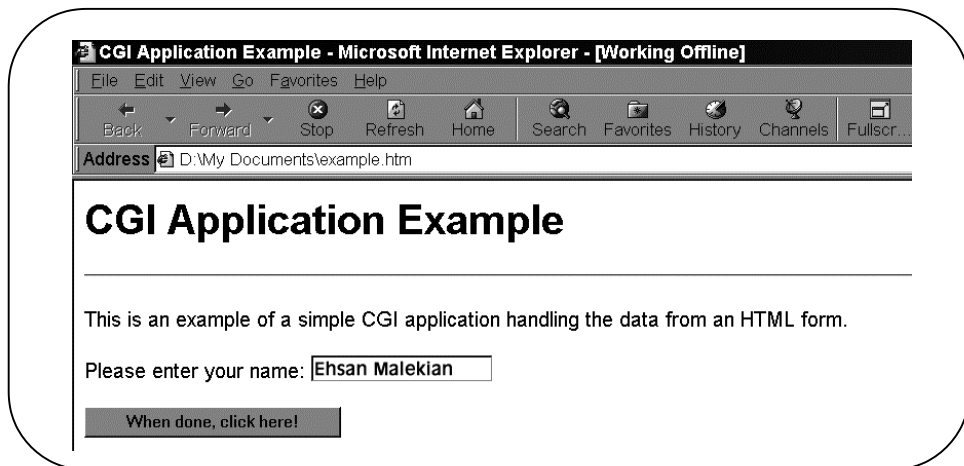
برای آشنایی با روش برنامه‌نویسی CGI ارائه یک مثال بسیار راهگشا خواهد بود. به صفحه وب شکل (۱۸-۱۰) و فایل HTML آن دقت کنید.

^۱ برای درک این مفهوم تکنیک Piping در یونیکس و مفهوم کنسول (Consol I/O) را به یاد بیاورید.


```

<HTML>
<HEAD>
<TITLE>CGI Application Example</TITLE>
</HEAD>
<BODY>
<H1>CGI Application Example</H1>
<hr>
This is an example of a simple CGI application
handling the data from an HTML form.
<BR>
<FORM ACTION="http://www.hqz.com/scripts/cgisamp.exe" METHOD="Post">
Please enter your name: <INPUT NAME="name" TYPE="text"><p>
<input type="submit" value="When done, click here!">
</FORM>
</BODY>
</HTML>

```



شکل (۱۸-۱۰) یک صفحه وب فرضی برای فراخوانی برنامه CGI

وقتی کاربر "کلید ارسال" را فشار می‌دهد، سرویس‌دهنده برنامه‌ای به نام `cgisamp.exe` را فراخوانی کرده و ورودی را به کنسول `stdin` هدایت می‌نماید. در ضمن متغیرهای محیطی جدول (۱۷-۱۰) را نیز تنظیم می‌کند. (به غیر از `QUERY_STRING`) برنامه `cgisamp.exe` را که به زبان C نوشته شده، بررسی می‌نماییم. در این برنامه پنج تابع تعریف شده است:

← تابع `void strevrt(char *cStr, char cOld, char cNew)`: رشته `cStr` را جستجو کرده و هرگاه کاراکتر `cOld` را در آن بیابد به کاراکتر `cNew` تبدیل می‌کند.

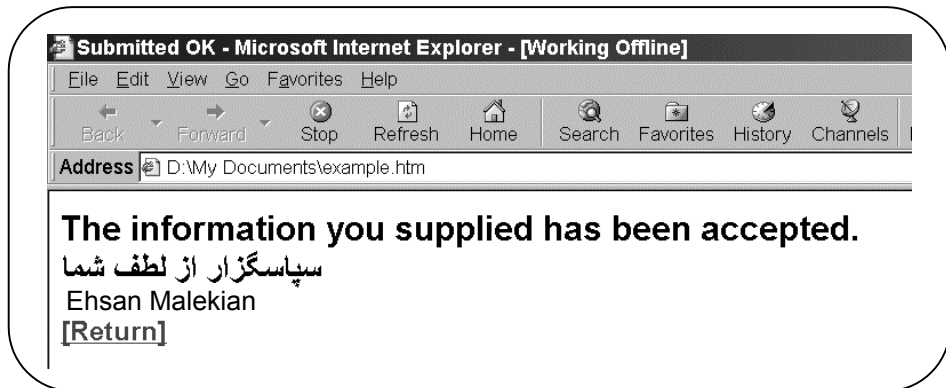
← تابع `static int TwoHex2Int(char *pC)`: اگر در یک رشته ورودی، کدهایی باشند که با علامت % و معادل هگزادسیمال آن مشخص شده باشند آنرا به کد اصلی بر می‌گرداند.

◀ تابع `void urlDecode(char *p)` : این تابع تمام رشته داده ارسالی از مرورگر را بررسی و به حالت اصلی تبدیل می‌کند.

◀ تابع `void StoreField(char *f, char *Item)` : این تابع زوج "نام فیلد/مقدار" را از رشته ورودی استخراج می‌کند.

◀ تابع اصلی برنامه (`main()`) : این برنامه داده‌های ارسالی توسط مرورگر را از `stdin` می‌خواند و بر اساس ورودی (نام کاربر) ، یک خروجی ساده طبق شکل (۱۹-۱۰) برای آن تولید و از طریق `stdout` برای مرورگر ارسال می‌کند.

```
<HEAD><TITLE>Submitted OK</TITLE></HEAD>
<BODY><h2>The information you supplied has been accepted.
<br>سیاسگزار از لطف شما<br>Ehsan Malekian</h2>
<h3><a href="http://www.hqz.com/cgisamp.htm">
[Return]</a></h3></BODY>
```



شکل (۱۹-۱۰) خروجی تولید شده توسط برنامه CGI

در ادامه اصل برنامه CGI که به زبان C نوشته شده ، ارائه شده است. این برنامه با صفحه وب شکل (۱۸-۱۰) تراکنش داشته و یک خروجی مطابق با شکل (۱۹-۱۰) تولید و ارسال می‌نماید. این برنامه با نام `cgisamp.exe` بر روی سرویس‌دهنده نصب می‌شود. این برنامه را در جدول (۲۰-۱۰) ملاحظه می‌کنید. اگر به دستورات `printf()` دقت کنید هیچ تفاوتی با برنامه‌نویسی معمولی زبان C ندارند؛ با این تفاوت که پیغامهای ارسالی روی خروجی ، متن معمولی نیستند بلکه با برجسبهای HTML غنی شده‌اند تا بر روی مرورگر نشان داده شوند.

```

/*****
• File: cgisamp.c
*
• Use: CGI Example Script.
*
• Notes: Assumes it is invoked from a form and that REQUEST_METHOD is POST.
• Ensure that you compile this script as a console mode app.
*
• This script is a modified version of the script that comes with EMWAC
• HTTPS.
*
• Date: 8/21/95
• Christopher L. T. Brown clbrown@netcom.com
*
*****/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <io.h>

char InputBuffer[4096];
static char * field;
static char * name;

/* Convert all cOld characters */
/* in cStr into cNew characters. */
void strcvrt(char *cStr, char cOld, char cNew)
{
    int i = 0;

    while(cStr[i])
    {
        if(cStr[i] == cOld)
            cStr[i] = cNew;
        i++;
    }
}

/* The string starts with two hex */
/* characters. Return an integer */
/* formed from them. */
static int TwoHex2Int(char *pC)
{
    int Hi, Lo, Result;

    Hi = pC[0];
    if('0' <= Hi && Hi <= '9')
        Hi -= '0';
    else if('a' <= Hi && Hi <= 'f')
        Hi -= ('a' - 10);
    else if('A' <= Hi && Hi <= 'F')
        Hi -= ('A' - 10);

    Lo = pC[1];
    if('0' <= Lo && Lo <= '9')
        Lo -= '0';
    else if('a' <= Lo && Lo <= 'f')

```

```

        Lo -= ('a' - 10);
    else if('A' <= Lo && Lo <= 'F')
        Lo -= ('A' - 10);

    Result = Lo + 16 * Hi;
    return(Result);
}

/* Decode the given string in-place */
/* by expanding %XX escapes.      */
void urlDecode(char *p)
{
    char *pD = p;

    while(*p)
    {
        if(*p == '%') /* Escape: next 2 chars are hex      */
        { /* representation of the actual character.*/
            p++;
            if(isxdigit(p[0]) && isxdigit(p[1]))
            {
                *pD++ = (char)TwoHex2Int(p);
                p += 2;
            }
        }
        else
            *pD++ = *p++;
    }
    *pD = '\0';
}

/* Parse out and store field=value items. */
/* Don't use strtok!                       */
void StoreField(char *f, char *Item)
{
    char *p;

    p = strchr(Item, '=');
    *p++ = '\0';
    urlDecode(Item);
    urlDecode(p);
    strcvrt(p, '\n', ' ');
    strcvrt(p, '+', ' '); /* Get rid of those nasty +'s */
    field = f;           /* Hold on to the field just in case. */
    name = p;           /* Hold on to the name to print*/
}

int main(void)
{
    int ContentLength, x, i;
    char *p,
        *pRequestMethod,
        *URL,
        *f;

    /* Turn buffering off for stdin.*/
    setvbuf(stdin, NULL, _IONBF, 0);

    /* Tell the client what we're going to send */

```

```

printf("Content-type: text/html\n\n");

/* What method were we invoked through? */
pRequestMethod = getenv("REQUEST_METHOD");

/* Get the data from the client */
if(strcmp(pRequestMethod,"POST") == 0)
{
    /* according to the requested method.*/
    /* Read in the data from the client. */
    p = getenv("CONTENT_LENGTH");
    if(p != NULL)
        ContentLength = atoi(p);
    else
        ContentLength = 0;
    if(ContentLength > sizeof(InputBuffer) -1)
        ContentLength = sizeof(InputBuffer) -1;

    i = 0;
    while(i < ContentLength)
    {
        x = fgetc(stdin);
        if(x == EOF)
            break;
        InputBuffer[i++] = x;
    }
    InputBuffer[i] = '\0';
    ContentLength = i;

    p = getenv("CONTENT_TYPE");
    if(p == NULL)
        return(0);

    if(strcmp(p, "application/x-www-form-urlencoded") == 0)
    {
        p = strtok(InputBuffer, "&");    /* Parse the data */
        while(p != NULL)
        {
            StoreField(f, p);
            p = strtok(NULL, "&");
        }
    }
}

URL = getenv("HTTP_REFERER");    /* What url called me.*/
printf("<HEAD><TITLE>Submitted OK</TITLE></HEAD>\n");
printf("<BODY><h2>The information you supplied has been accepted.");
printf("<br>سپاسگزار از لطف شما %s</h2>\n", name);
printf("<h3><a href='%s'>[Return]</a></h3></BODY>\n", URL);

return(0);
}

```

۶) مفهوم حقیقت مجازی^۱ -VR-

”حقیقت مجازی“ (که به اختصار VR گفته می شود) در یک مفهوم عام، به صورت زیر تعریف می شود:

”شبهه سازی فضای سه بعدی توسط تکنیکهای نرم افزارهای روی فضائی که ذاتاً دو بعدی است؛ بگونه ای که کاربر می تواند به غیر از مشاهده این فضا، در آن سیر کرده و در محیط تغییراتی را نیز ایجاد نماید.“

مفهومی که ارائه شد ذاتاً دارای ابهام است؛ زیرا هر کس می تواند از آن استنتاج شخصی خود را داشته باشد. برای روشنتر شدن قضیه، صفحه شطرنجی را در نظر بگیرید که از روبرو به آن می نگرید. در محیط VR شما می توانید این صفحه را طبق تمایلتان بچرخانید و از زوایای مختلف به آن نگاه کنید؛ در عین حال اشیاء صفحه (مهره ها) را جابجا کنید (با استفاده از صفحه کلید، موس، دسته فرمان^۲) و در نهایت نتیجه را روی صفحه ببینید. چنین کاری با تصاویر دو بعدی گرفته شده از طبیعت امکان پذیر نیست، زیرا شما ملزم به دیدن تصویر از زاویه ای هستید که دوربین آن را ثبت کرده است. قابلیت تغییر اشیاء در محیط VR به این دلیل است که هر شیئی در این محیط مشخصات و مختصات فضای سه بعدی خود را دقیقاً حفظ می کند؛ ولی چون صفحه نمایش دو بعدی است، بعد سوم با استفاده از روشهای پرسپکتیو روی فضای دو بعدی نگاشته می شود.^۳ چون اشیاء در محیط VR مختصات فضایی (x,y,z) خود را حفظ می کنند، بنابراین با تغییر زاویه دید یا چرخاندن صفحه، دور یا نزدیک شدن به اشیاء، عمل نگاشت بُعد z روی صفحه x-y از نو انجام می شود؛ همانند عملی که در محیطهای واقعی برای چشم اتفاق می افتد. - شکل (۲۱-۱۰) - اگر هنوز در مفهوم حقیقت مجازی (VR) شک دارید یا به یقین نرسیده اید تا انتهای فصل صبر کنید.

ممکن است حقیقت مجازی در ذهن شما محیط های چندرسانه ای^۴ را تداعی کند و این چندان عجیب نیست چرا که هر دوی این محیطها شامل صدا و تصویر هستند و هر دوی آنها تقریباً با هم رشد کرده اند و پیدایش آنها در دنیای ”تکنولوژی اطلاعات“^۵ سبب ایجاد تحولات شگرف شده است؛ ولیکن تفاوت های بنیادی بین محیطهای حقیقت مجازی (VR) و محیطهای

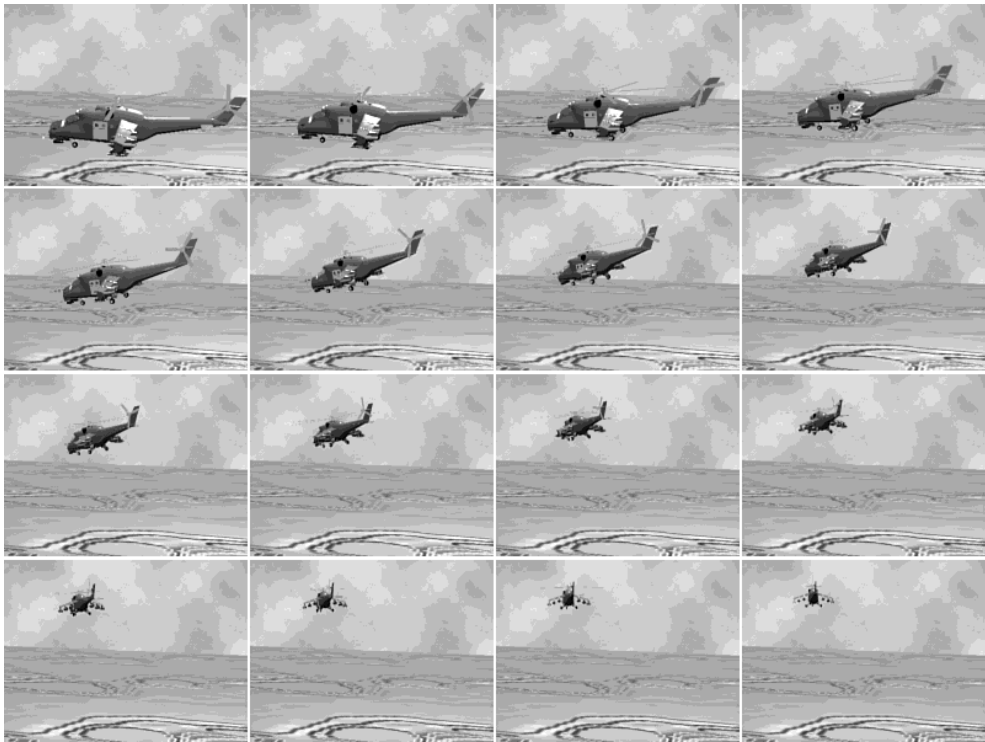
^۱ Virtual Reality

^۲ Joystick

^۳ Mapping

^۴ Multimedia

^۵ Information Technology



شکل (۲۱-۱۰) تعریف یک شیء در محیط VR و نمایش آن از زوایای مختلف

چندرسانه‌ای وجود دارد. این تفاوتها را می‌توان در موارد کلی زیر خلاصه کرد:

◀ محیط‌های چندرسانه‌ای با داده‌هائی سروکار دارند که از قبل جمع آوری و برنامه‌ریزی شده‌اند. اینها در حقیقت یکسری اطلاعات دسته‌بندی و ذخیره شده هستند که به صورت متوالی و با یک روال از قبل مشخص در یک فضای دو بعدی به نمایش درمی‌آیند. فیلم، تصاویر متحرک (به همراه صدا) و یا هر چیزی در دنیای چندرسانه‌ای، یکسری تصاویر دو بعدی ذخیره شده هستند که طبق قاعده خاص و هماهنگ (و شاید مقداری پیش‌پردازش) روی صفحه نمایش ظاهر می‌شوند. برخلاف محیط‌های چندرسانه‌ای، اشیاء در محیط VR، طبیعتشان سه‌بعدی است و علاوه بر مختصات طول و عرض، پارامتر ارتفاع را هم دارند.

◀ در محیط‌های چندرسانه‌ای کاربر نمی‌تواند آنچه را که وجود دارد، تغییر بدهد یا چیزی به آن اضافه نماید. مثلاً وقتی که یک فیلم یا انیمیشن را نگاه می‌کنید، نمی‌توانید زاویه دیدتان را تغییر بدهید و یا صفحه نمایش را بگونه‌ای بچرخانید که از پشت به تصاویر نگاه کنید.

برخلاف محیط‌های چندرسانه‌ای، محیط‌های حقیقت مجازی کاملاً محاوره‌ای است و قابلیت سازگاری و تغییر دائم دارد و کاربر می‌تواند اشیاء محیط را جابجا کند، یا آنها را از زوایای مختلف نگاه کند و در یک کلام می‌تواند خود را در یک محیط واقعی احساس کند.

در تکنولوژی VR، کاربر می‌تواند با استفاده از کلاه‌های مخصوصی بنام HMD^۱ یا عینک‌های ویژه، خود را دقیقاً در محیط VR احساس نماید؛ در آن بگردد و با آن در تعامل باشد.

۱-۶) VRML^۲؛ (زبان مدل‌سازی حقیقت مجازی)

VRML زبانی است که توسط آن، هر شیئی در محیط VR با علائم مخصوص و گرامر خاص یک زبان نشانه‌گذاری^۳ بصورت استاندارد، مدل می‌شود. یعنی VRML قالب استاندارد فایلی است که درون آن اطلاعات مربوط به اشیاء محیط VR، بصورت متنی تعریف می‌شود.

برای نمایش یک فایل متنی VRML (که در آن مجموعه‌ای از اشیاء تعریف شده است) دو مرحله لازم است:

◀ خواننده شدن فایل متنی VRML، استخراج تک تک اشیاء و پارامترهای آن و تبدیل آن به یک "ساختمان داده" مناسب برای تفسیر و نمایش. این قسمت از کار را برنامه‌روگر بر عهده دارد.

◀ تحویل دادن ساختمان داده تشکیل شده برای اشیاء به برنامه‌ای که آنرا پردازش و تفسیر کرده و نهایتاً نمایش می‌دهد. به این برنامه "موتور تفسیر"^۴ گفته می‌شود.

تمام جنبه‌های دنیای مجازی و فعل و انفعالاتی که کاربر می‌تواند با این محیط داشته باشد توسط استاندارد VRML تعریف می‌شود. فایل‌های VRML را می‌توان با فایل‌های HTML مقایسه کرد زیرا هر دو فایل‌های متنی و استاندارد هستند که توسط مرورگر خوانده و برای نمایش، تفسیر می‌شوند.

زبان VRML به این دلیل طراحی شد تا بتوان با فایل‌های بسیار کم حجم، صحنه‌های سه‌بعدی طراحی شده در محیط VR را مستند کرده و آنها را روی شبکه

^۱ Head Mounted Display

^۲ Virtual Reality Modeling Language

^۳ Markup Language

^۴ Rendering Engine

اینترنت (که فعلاً از محدودیت پهنای باند رنج می برد) ارسال کرد. در حقیقت VRML مشخصات لازم جهت تفسیر و نمایش اشیاء هر صفحه VR را مدلسازی کرده و این فایل بر روی شبکه اینترنت انتقال می یابد و پس از دریافت آن توسط ماشین کاربر، تفسیر شده و نمایش داده خواهد شد.

فلسفه وجودی VRML و HTML از یک منبع سرچشمه گرفته است؛ زیرا بعنوان مثال بارشدن یک تصویر دوبعدی کامل از یک صحنه بر روی ماشین کاربر اینترنت که از خط تلفن بعنوان خط ارتباط استفاده می کند، ۲۰ دقیقه طول می کشد. در حالی که اگر همان صحنه را توسط VRML مدلسازی نمائیم، بسته به پیچیدگی اشیاء آن در چند ثانیه بار می شود.^۱ البته باید این نکته را خاطرنشان کرد که بنابر طبیعت اشیاء در محیط VR، فایل های VRML از فایل های HTML بزرگتر هستند و حجم پردازشی که در ماشین کاربر باید برای تفسیر و نمایش یک فایل VRML انجام شود، هزاران برابر حجم پردازش برای نمایش یک فایل HTML است. بنابراین کاربر برای آنکه بتواند در حین بار شدن فایل VRML، اشیاء را در محیط VR بصورت بلادرنگ^۲ ببیند، نیاز به سخت افزار و مرورگر بسیار سریع دارد.^۳

به هر حال VRML هنوز به طور کامل رشد نکرده و حتی به عقیده برخی فعلاً شکست خورده است ولی شاید با گذشت زمان کاربردهای زیبای آن فراگیر شود.^۴ بعنوان مثال شاید در آینده شما بتوانید برای خرید یک خانه از طریق اینترنت، به سایت یک بنگاه معاملات ملکی وصل شوید، خانه ای را انتخاب کنید و با بارکردن فایل VRML آن، درب خانه مورد نظرتان را باز کنید، وارد آن شوید، در اتاقها بچرخید و زوایای مختلف آنرا نگاه کنید و حتی چشم انداز بیرون را تماشا کنید.

^۱ به عنوان مثال یک صفحه شطرنج با تمام مهره های آن و عملیاتی که کاربر می تواند بر روی آن انجام بدهد، توسط نرم افزار Caligari Pioneer حدود ۷۵ کیلوبایت است که انتقال آن به مرورگر کاربری که با مودم 33.6kbps به اینترنت متصل شده است، زیر بیست ثانیه طول می کشد.

^۲ Real Time

^۳ یکی از این مرورگرها که فایل های VRML را تفسیر می کند، Live3D نام دارد.
^۴ دلایل عدم پیشرفت VRML را می توان نیاز به تخصص و هنر کافی برای طراحی اشیاء سه بعدی و همچنین سخت افزار گرانتر عنوان کرد. طراحی یک صحنه VR، صدها برابر طراحی صفحات HTML، نیاز به هزینه و وقت دارد که اکثریت نمی توانند چنین هزینه ای را متقبل شوند.

۶-۴) اصول VRML

در بخش قبل اشاره شد که VRML فایللی است متنی و تبدیل آن به تصویر ، شامل چند مرحله است که در این قسمت آنرا بیشتر تشریح خواهیم کرد: پس از بار شدن یک فایل VRML ، مرورگر آنرا پویش کرده و مجموعه‌ای از ساختمان داده‌ها را در حافظه تولید می‌کند. ("ساختمان داده" مجموعه‌ای از اطلاعات سازمان‌یافته در حافظه کامپیوتر است که برای هر شیئی تشکیل می‌شود). پس از تشکیل ساختمان داده‌های لازم ، مرورگر "موتور تفسیر" را برای نمایش اشیاء فرا می‌خواند. موتور تفسیر دو وظیفه برعهده دارد:

- ◀ محاسبات و پردازش لازم
- ◀ رسم تصاویر

موتور تفسیر را می‌توان پروسه‌ای در نظر گرفت که کارش رسم تصاویر محیط VR با استفاده از ساختمان داده حاصل از پویش فایل VRML می‌باشد. همانگونه که اشاره شد گاه حجم محاسبات لازم برای تشکیل و ترسیم تصاویر محیط VR بسیار زیاد است ، بهمین دلیل برای بالا بردن سرعت بخشی از این پردازش به سخت‌افزار کارتهای گرافیکی ویژه محول می‌شود.

موتورهای تفسیر غالباً از مفاهیم شیئی‌گرائی^۱ و ذخیره‌بُرداری^۲ برای ساختمان داده‌های مربوط به هر شیئی در محیط VR استفاده می‌کنند. برای مشخص شدن مفهوم ذخیره‌سازی بُرداری ، دستگاه مختصات قائم را با یک مرکز فرضی در فضای سه بعدی داشته در نظر بگیرید. به هر نقطه که در فضای سه‌بعدی با مختصات x-y-z بصورت یکتا مشخص می‌شود ، بُردار می‌گوییم. یک پاره خط در محیط VR می‌تواند فقط با دو بردار (x_0, y_0, z_0) و (x_1, y_1, z_1) مشخص شود. برای رسم یک چند ضلعی در فضا به بردارهای مربوط به هر رأس نیاز است. اشیاء شناخته‌شده‌ای مثل استوانه ، کره و مخروط با استفاه از روابط ریاضیشان ، در دستگاه مختصات سه‌بعدی قابل ترسیم هستند. بعنوان مثال برای رسم یک کره در فضای سه بعدی ، دو پارامتر شعاع کره و بُردار مرکز آن نیاز است . موتور تفسیر براساس رابطه ریاضی زیر نقاط سطح کره را محاسبه و ترسیم می‌نماید:

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2$$

^۱ Object Oriented
^۲ Vector

هر شیئی خواه ساده خواه پیچیده می تواند براساس اشیاء ساده تر یا مجموعه ای از بردارها توصیف شود. برای آنکه اشیاء بتوانند تغییر کنند می توان عملیات زیر را روی آنها تعریف کرد:

◀ **عمل مقیاس (Scaling):** این عمل با یک ضریب شیئی را در جهت محورهای مختصات منقبض یا منبسط می کند. عمل مقیاس، با رابطه ریاضی زیر مدل می شود:

$$(x, y, z) \rightarrow (\alpha x, \beta y, \gamma z)$$

اگر α ، β یا γ بزرگتر از ۱ باشند، شیئی منبسط و اگر کوچکتر از ۱ باشد منقبض می شود.

◀ **عمل چرخش (Rotation):** چرخش یک شیئی حول یک بردار

◀ **عمل انتقال (Translation):** انتقال یک شیئی از یک مکان به مکان جدید.

حرکات یا تغییرات پیچیده یک شیئی، می تواند بر اساس ترکیبات سه عمل اساسی بالا توصیف شود. این عملیات بر اساس محاسبات ضرب ماتریسی (بعبارت عام تر، محاسبات جبر خطی) توسط CPU انجام می شود.

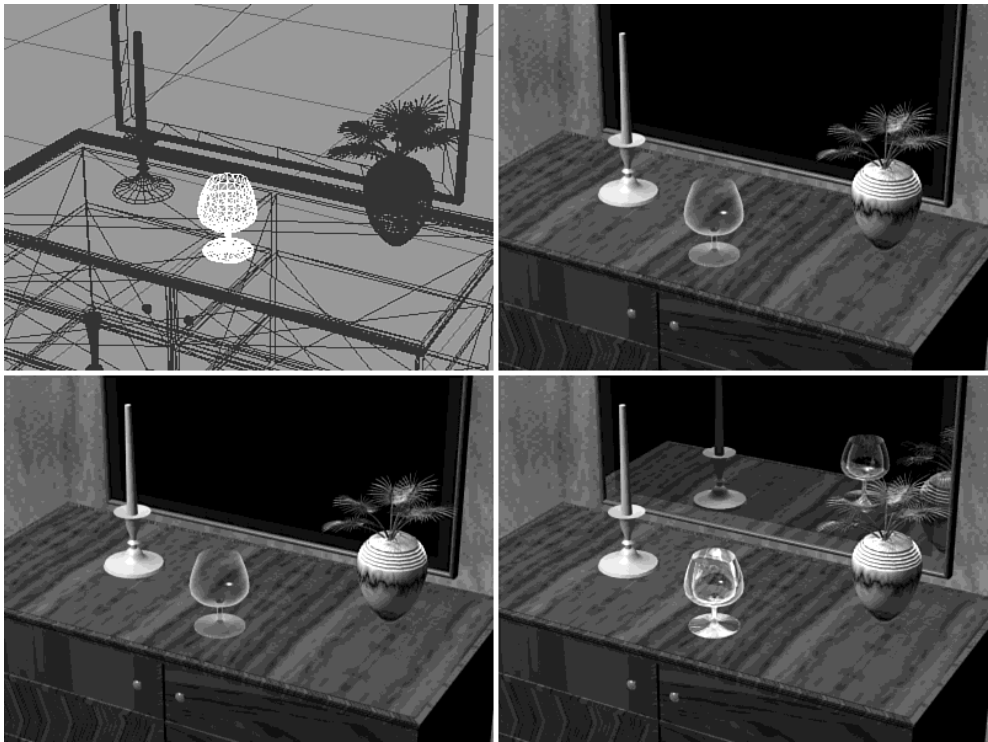
عملیاتی که ذکر شد بصورت مجزا بر روی تک تک اشیاء یک صحنه قابل انجام است، ولی نکته اساسی اثرات و تعاملاتی است که اشیاء یک صحنه بر روی یکدیگر خواهند داشت. مثلاً با نگاه از روبرو به کتابی که در جلو یک مداد قرار گرفته است، چون مداد پشت کتاب پنهان شده، نباید از این زاویه دیده شود؛ ولیکن اگر کاربر صحنه را بچرخاند، مداد دیده خواهد شد. یک لیوان یا یک جسم شفاف روی یک صحنه، باید اشیاء پشت خود را نشان دهد. یک آینه باید تصویر اشیاء روبروی خود را نشان بدهد. در شکل (۲۲-۱۰) تعامل اشیاء با یکدیگر در محیط VR نشان داده شده است.

موتور تفسیر برای ترسیم هر شیئی به پردازشهای زیر نیاز دارد:

◀ بررسی موقعیت یک شیئی نسبت به اشیاء دیگر

◀ محاسبه درخشندگی شیئی: موتور تفسیر باید شدت درخشندگی هر نقطه در تصویر و تغییرات در مقدار روشنائی یک شیئی را در اثر مجاورت با اشیاء دیگر، محاسبه نماید.

◀ محاسبه سطوح قابل رؤیت: در هنگام ترسیم شیئی باید فقط سطوحی نشان داده شوند که از زاویه دید فعلی، قابل رؤیت هستند. محاسبه سطوح قابل رؤیت برعهده موتور تفسیر است.



شکل (۲۲-۱۰) تعامل اشیاء با یکدیگر در محیط VR

محاسبه پرسپکتیو شیء: با توجه به آنکه صفحه نمایش تخت و دو بعدی است ولی تصاویر توصیف شده در فایل VRML ذاتاً سه بعدی هستند، موتور تفسیر باید به نحوی مختصه Z از هر شیئی را روی صفحه دو بعدی x-y بنگارد، تا جسم دوبعدی توسط چشم، سه بعدی تجسم شود.

محاسبات ذکر شده فقط شمه‌ای از عملیات لازم بر روی اشیاء است. پردازشهای متنوع دیگری هم وجود دارند تا محیط VR را طبیعی‌تر جلوه دهند. البته این محاسبات بسیار وقتگیر هستند.

۳-۶) ساختار یک فایل VRML

فایل‌های VRML در حقیقت مجموعه‌ای از اشیاء تعریف شده به زبان VRML (در محیط گرافیک سه بعدی) هستند و همانند فایل‌های HTML ذاتاً فایل‌های متنی ساده ASCII محسوب می‌شوند که براحتی در یک ویرایشگر متن، قابل ایجاد و ویرایش

هستند. فایل‌های VRML پس از ایجاد، معمولاً با پسوند .wrl ذخیره می‌شوند. (پسوندهای .vrm یا .vrm هم استفاده شده است.)

وقتی مرورگر یک فایل را از سرویس دهنده وب بار می‌کند، ابتدا نوع فایل و چگونگی تفسیر آنرا تعیین می‌کند. انتقال این فایل دقیقاً مبتنی بر پروتکل HTTP بوده و هیچ تفاوتی با انتقال فایل‌های HTML ندارد. تنها تفاوت در نوع مرورگری است که موظف به تفسیر اینگونه فایل‌هاست. نسخه‌های جدید مرورگر Netscape بصورت داخلی قادرند فایل‌های VRML را پویا کرده و تفسیر نمایند ولی مرورگرهای فعلی مایکروسافت (مثل IE5.0) نیاز به "برنامه‌های اتصال" دارند.

یک فایل VRML از چهار قسمت تشکیل شده است:

◀ **خط سرآیند فایل:** اولین خط در هر فایل VRML، با علامت # شروع می‌شود و مشخصات فایل را تعیین می‌کند. این خط قالبی همانند مثال زیر دارد:

```
#VRML V1.0 ascii
```

یعنی محتوای فایل از نوع VRML و با کدهای استاندارد ASCII است و مطابق با نسخه 1.0 از زبان VRML تدوین شده است. یا:

```
#VRML V2.0 utf2
```

محتوای فایل از نوع VRML و با کدهای ASCII (شامل کدهای بالاتر از ۱۲۸) است و مطابق با نسخه 2.0 از زبان VRML تدوین شده است.

◀ **خطوط توضیح:** به غیر از خط اول از فایل که با علامت # شروع می‌شود، هر خطی که در متن فایل با این علامت شروع شود به عنوان خط توضیح تلقی شده و نادیده انگاشته می‌شود.

◀ **گره‌ها:** یک فایل VRML از عناصری تشکیل شده است که به آن "گره" گفته می‌شود. گره می‌تواند معرف یک شیء ساده یا یک شیء پیچیده سه‌بعدی باشد. یک گره در فایل VRML بصورت زیر نمایش داده می‌شود:

```
NodeType { Fields }
```

◀ **فیلدها:** در تعریف یک گره به صورت بالا، مقادیری که بین {...} قرار می‌گیرد، توصیف آن گره است و چگونگی تفسیر یا نمایش یک شیء را تعیین می‌نماید. به مثال زیر دقت نمایید:

#VRML V1.0 ascii	خط سرآیند
#This file defines a simple red sphere	خط توضیح
Separator {	یک گره (شیء کلی صحنه)
Material { diffuseColor 1 0 0 } #The color red	یک گره (تعریف رنگ پوسته اشیاء)
Sphere { }	یک گره (تعریف کره‌ای با شعاع ۱ در مرکز)
}	پایان تعریف گره (پایان تعریف شیء صحنه)

اگر به دنیای واقعی اطرافتان نگاه کنید، مجموعه‌ای از اشیاء سه‌بعدی با مشخصات خاص خود را می‌بینید. این اشیاء معمولاً ساختاری پیچیده دارند و از اشیاء ساده و کوچکتر تشکیل شده‌اند. مثلاً یک میز را به عنوان شیئی مستقل در فضای سه‌بعدی در نظر بگیرید. این شیء را می‌توان به پنج شیء کوچکتر تقسیم کرد (۴ پایه و یک سطح). اگر میز را "شیء پدر" در نظر بگیریم، اجزای کوچکتر آن "فرزند" تلقی می‌شوند. اشیاء فرزند می‌توانند خصوصیات پدر را به ارث ببرند یا آنکه خصوصیات متمایز از پدر خود داشته باشند. مثلاً پایه میز به عنوان فرزند می‌تواند رنگ خود را از شیء پدر به ارث ببرد. اشیاء فرزند می‌توانند خودشان دارای اجزای کوچکتری به عنوان فرزند باشند. هرگونه تغییر در موقعیت شیء پدر باید تمام فرزندان را تحت تاثیر قرار بدهد.

VRML دقیقاً مبتنی بر مفهوم شیء‌گرایی^۲ و ارث‌بری^۳ است. برای شروع، به تعریف بدن انسان در VRML دقت کنید:

```
#VRML V1.0 ascii
# A simple humanoid body made up of cylinders and spheres
# Built by hand with a text editor!
```

^۱ Child
^۲ Object Oriented Concept
^۳ Inheritance

```
DEF Body Separator {
  DEF Pelvis Separator {
    DEF Chest Separator {
      DEF LeftUpperArm Separator {
        DEF LowerArm Separator {
          #تعریف اجزا در اینجا می آید#
        }
      }
    }
    DEF RightUpperArm Separator {
      DEF LowerArm Separator {
        #تعریف اجزا در اینجا می آید#
      }
    }
  }
  DEF Head Separator {
    DEF Skull Separator {
      #تعریف اجزای مجسمه در اینجا می آید#
    }
    DEF LeftEye Separator {
      #تعریف اجزای چشم چپ در اینجا می آید#
    }
    DEF RightEye Separator {
      #تعریف اجزای چشم راست در اینجا می آید#
    }
    DEF Nose Separator {
      #تعریف اجزای بینی در اینجا می آید#
    }
  }
  DEF LeftThigh Separator {
    DEF Calf Separator {
      #تعریف اجزای ساق پای چپ در اینجا می آید#
    }
  }
  DEF RightThigh Separator {
    DEF Calf Separator {
      #تعریف اجزای ساق پای راست در اینجا می آید#
    }
  }
}
```

در این مثال اگر شیء "بدن" را به عنوان "پدر" تلقی کنیم، اشیایی مثل دست، سر، بازو و ... فرزندان آن محسوب می‌شوند. هر شیء فرزند مثل "دست" خود شامل اشیاء فرزند مثل انگشتان، بازو و ساعد است. هر گونه تغییر و تبدیل شیء پدر (بدن) باید به اشیاء فرزند نیز اعمال شود. به عنوان مثال حرکت بدن به سمت جلو یا چرخش آن باید منجر به حرکت یا چرخش یکایک اجزای آن نیز بشود. (در مثال بالا تعریف جزئیات هر شیء فرزند نیامده است.)

قالب یک شیء در VRML با گره **Separator** تعریف می‌شود. در تعریف گره‌ها با Separator مشخصه‌های زیر می‌تواند روی نحوه نمایش اشیاء در صحنه VR تاثیر بگذارد:

Separator {

Transformations Definition

Surface properties

Shapes

Children

}

اشیاء فرزند نیز دقیقاً همانند پدر خود با ساختار Separator تعریف می‌شوند و هرگاه مشخصاتی مثل تبدیل یا رنگ سطح آن تعریف نشود، آنرا از پدر خود به ارث خواهند برد.

هر فایل VRML دارای یک شیء کلی به عنوان پدر تمام اشیاء موجود در صحنه VR می‌باشد و هر شیء دیگر در درون آن تعریف می‌شود. به عبارت دیگر این شیء، **ظرفی**^۱ برای بقیه اشیاء محسوب می‌شود. به یک مثال ساده دقت کنید:

Separator {

Material { diffuseColor 0 1 0 }

Transform { translation 2 5 7 }

Cone { }

Material { diffuseColor 0 1 0 }

Transform { translation 2 5 7 }

Sphere { }

}

^۱ Container

در این مثال در شیئی پدر که شامل کل صحنه VR می باشد ، ابتدا رنگ سطح اشیاء ، سبز (R=0,G=1,B=0) نظر گرفته می شود.^۱ (با گره { Material }) . سپس یک عمل انتقال به اندازه (x=2,y=5,z=7) تعریف می شود. تمام اشیاء بعدی نسبت به موقعیت جدید ترسیم خواهند شد؛ (با گره { Transform }) . در ادامه ، یک مخروط با مشخصات پیش فرض تعریف شده است؛ (با گره { Cone }) . بعد از آن رنگ سطح اشیاء ، به قرمز (R=1,G=0,B=0) تبدیل شده و یک عمل انتقال جدید تعریف می شود. تمام اشیاء بعدی نسبت به موقعیت جدید ترسیم می شوند ، سپس یک گره با مشخصات پیش فرض تعریف شده است؛ (با گره { Sphere }) .

گره { Material } اثر قبلی خود را باطل^۲ می کند ، در حالی که عمل انتقال بعدی توسط گره { Transform } ، نسبت به انتقال قبلی محاسبه می شود؛ یعنی گره قرمز رنگ به مرکز (x=10,y=14,z=17) نمایش خواهد یافت. (x=8+2,y=9+5,z=17)

هرگاه یک شیئی درون شیئی دیگر تعریف شود ، فرزند آن محسوب می شود در حالی که اگر بیرون از آن تعریف گردد ، مستقل از دیگر فرزندان خواهد بود. مثال :

Separator {	شیئی پدر
Separator {	شیئی فرزند
Separator {	شیئی فرزند فرزند (نوه)
}	
}	
}	

Separator {	شیئی پدر
Separator {	شیئی فرزند
}	
Separator {	شیئی فرزند
}	
}	

^۱ رنگ در VRML با سه مولفه RED/GREEN/BLUE تعریف می شود. هر یک از مقادیر RGB عددی اعشاری بین صفر و یک هستند.

^۲ Override

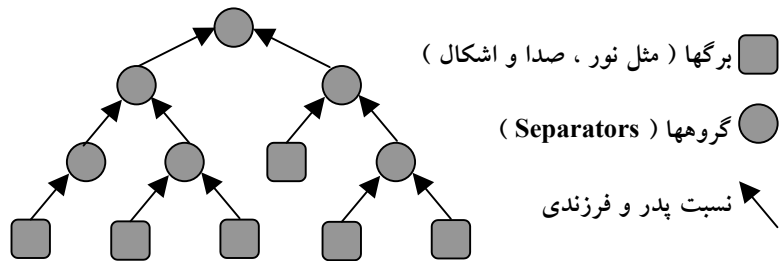
در VRML لازم نیست تمام اشیاء دارای ظاهر گرافیکی یا هندسی باشند بلکه یک شیء ممکن است فقط برای آن بوجود بیاید که دو شیء دیگر را به عنوان اشیاء فرزند خود، در یک گروه قرار بدهد.

بنیان ساختمان داده اشیاء در VRML درخت معکوس است که سلسله‌مراتبی از گره‌ها را تعریف می‌کند. در شکل (۲۳-۱۰) این ساختار سلسله‌مراتبی به تصویر کشیده شده است. دو نوع اصلی شیء وجود دارد:

◀ برگ (Leaf)

◀ گروه (Group)

هر گروه از گره‌ها می‌تواند شامل برگ و گروه‌های دیگری باشد. برگ‌ها معمولاً مطابق با ترتیب اشیایی است که شما آنها را در فضای سه‌بعدی می‌بینید. (نظیر اشکال، صدا، نور و ...)



شکل (۲۳-۱۰) ساختار سلسله‌مراتبی اشیاء در محیط VR

تعریف اشیاء در VRML تحت تاثیر مستقیم تجربه شما از جهان مجازی است؛ هر چیز دیدنی یا شنیدنی مثل یک میز، صندلی یا صدای باران بوسیله گره نشان داده می‌شود. صدای یک ساعت بوسیله گره صدا^۱ ایجاد می‌شود و در یک منظره دیدنی، یک یا چند گره مولد مجازی نور، صحنه را نورپردازی می‌کند. شما وقتی در دنیای VRML هستید نمی‌توانید گره گروه را ببینید، اما آنها وجود دارند و بر موقعیت و نحوه مشاهده گره‌های برگ در یک درخت تاثیر می‌گذارند. توجه کنید که در VRML 2.0 فرزندان همسطح معمولاً بی‌ارتباط با یکدیگر هستند. در این نسخه از زبان VRML حدود ۵۴ نوع متفاوت از گره‌ها وجود دارد. البته امکان تعریف

^۱ Sound Node

گره‌های جدید به صورت الگو^۱ وجود دارد که هر یک از این گره‌ها برای مشخص کردن یک شیئی خاص است که برای ایجاد یک صحنه در چند جا باید ظاهر شود.

هر گره دارای مجموعه‌ای از فیله‌های مقدار است؛ برای مثال یک گره از نور، فیله‌ی دارد که شدت نور را مشخص می‌کند و اگر ارزش آن فیله را تغییر بدهید، شدت روشنایی نور صحنه و اشیایی که در معرض آن هستند، تغییر خواهد کرد.

۴-۶) پیاده‌سازی گره‌ها در VRML

بار دیگر ساختار یک گره را بررسی می‌کنیم:

Separator {

Transformations Definition

Surface properties

Shapes

Children

}

اگر در یک گره هیچ شکل^۲ هندسی تعریف نشده باشد، هیچگونه نمود ظاهری نخواهد داشت. در ادامه فیله‌های یک گره را بررسی می‌کنیم:

۴-۶-۱) تبدیلات

در یک گره فیله‌ی {..} Transform، مجموعه‌ی تبدیلاتی را تعریف می‌کند که بر یک شیئی و فرزندانش اعمال می‌شود. این مجموعه‌ی تبدیلات می‌تواند بصورت زیر تعریف شود:

Transform {

Center *x y z*

scalingFactor *x y z*

scaleOrientation *x y z a*

rotation *x y z a*

translation *x y z*

}

^۱ Prototype
^۲ Shape

هر فیلد بر روی خط مجزا قرار می‌گیرد و رعایت ترتیب نوشتن آنها مهم نیست زیرا اجرای گره Transform توسط "موتور تفسیر" به ترتیب زیر خواهد بود:

۱. مقیاس (Scaling)
۲. چرخش (Rotation)
۳. انتقال (Translation)

قالب بالا تمام تبدیلات را در برمی‌گیرد و هر گاه به یک یا تعدادی از آنها نیاز نبود، فیلد مربوطه حذف می‌شود. مثال:

```
Transform {
    Center 3 1 2
    rotation 4 0 0 0.31
}
Transform {
    translation 4 0 3
}
```

◀ **Center**: نقطه‌ای را در فضای سه‌بعدی تعریف می‌کند که عملیاتی نظیر چرخش اشیاء حول آن انجام خواهد شد.

◀ **scalingFactor**: ضرائب انقباض یا انبساط اشیاء را در راستای سه محور مختصات تعریف می‌نماید.

◀ **scaleOrientation**: با این تبدیل ابتدا محور مختصات به اندازه a حول مرکز پیش‌فرض می‌چرخد و سپس در سه محور مختصات، با ضرائب x y z منقبض یا منبسط می‌شود.

◀ **rotation**: این تبدیل اشیاء را به اندازه a رادیان حول بردار $(x \ y \ z)$ می‌چرخاند.

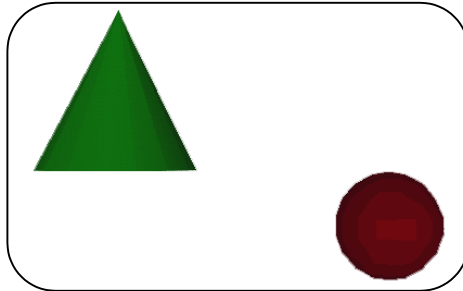
مثال زیر یک چرخش ۴۵ درجه‌ای حول محور y و سپس یک انتقال به اندازه نیم واحد^۱ (۰/۵ متر)، روی محور x تعریف می‌کند؛ (دقت کنید زاویه بطور تقریبی بر حسب رادیان نوشته شده است):

```
Transform {
    rotation 0 1 0 0.78541
    translation 0.5 0 0
}
```

در مثالهای بعد نمایش واقعی از اشیائی را که تحت تاثیر "گره تبدیل" قرار گرفته‌اند، ملاحظه می‌کنید:

^۱ معمولاً واحد طول در VRML، متر است که برای نمایش در یک پنجره نرمالیزه می‌شود.

```
#VRML V1.0 ascii
Separator {
  Separator {
    Material { diffuseColor 0 1 0 }
    Transform { translation 2 5 7
                  rotation 1 0 0 0.78
                }
  }
  Cone { }
}
Separator {
  Material { diffuseColor 1 0 0 }
  Transform { translation 6 3 2
                }
  Sphere { }
}
}
}
```



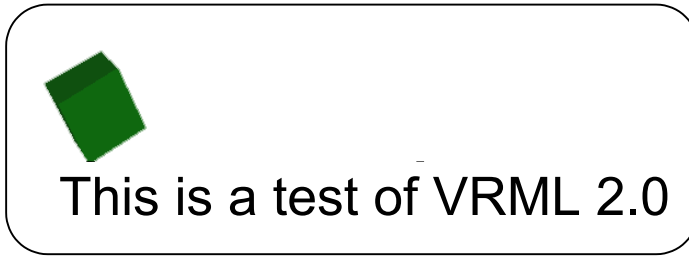
شکل (۲۴-۱۰) تاثیر تبدیلات بر روی اشیاء در محیط VR

```
#VRML V1.0 ascii
Separator {
  Separator {
    Material { diffuseColor 0 1 0 }
    Transform { translation 15 15 19 }
    Transform { rotation 2 2 2 0.78 }
    Cube {
      width 10
      height 10
      depth 10 }
  }
}
```

```

}
Separator {
  AsciiText {
    String [" This is a test of VRML 2.0"]
  }
}
}

```



شکل (۲۵-۱۰) تاثیر تبدیلات بر روی اشیاء در محیط VR

۶-۴-۲) اشکال هندسی در VRML

در VRML براحتی می توان اشکال شناخته شده هندسی را تعریف کرد. برخی از این اشیاء را معرفی می کنیم:

◀ کره : ایجاد کره بسیار ساده است و فقط نیاز به تعریف یک فیلد دارد:

```

Sphere {
  radius r          شعاع کره بر حسب متر
}

```

کره به مرکز پیش فرض ، ترسیم خواهد شد و اگر شعاع کره مشخص نشود مقدار آن ۱ فرض خواهد شد.

◀ مخروط :

```

Cone {
  bottomRadius r    شعاع قاعده مخروط بر حسب متر
  height h         ارتفاع مخروط بر حسب متر
}

```

اگر هر یک از فیلدها تعریف نشود ، مقادیر پیش فرض آنها به شرح ذیل است:
 شعاع قاعده : ۱ متر ارتفاع مخروط : ۲ متر مرکز ترسیم : مرکز پیش فرض

استوانه :

```
Cylinder {
    bottomRadius r      شعاع قاعده استوانه بر حسب متر
    height h           ارتفاع استوانه بر حسب متر
}
```

مکعب :

```
Cube {
    width w           طول مکعب بر حسب متر
    height h         ارتفاع مکعب بر حسب متر
    depth d          عمق مکعب بر حسب متر
}
```

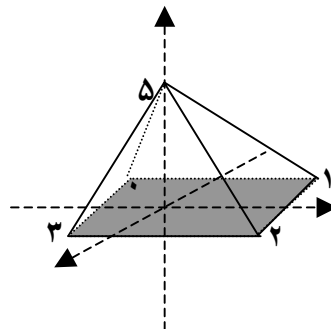
متن :

```
AsciiText {
    string s          رشته متنی مورد نظر برای نمایش
    spacing sp       ضرب فاصله عمودی سطرهاى متن از هم
    justification j   موقعیت نمایش متن ( وسط ، چپ یا راست )
}
```

اشکال هندسی مرکب : گره‌های هندسی که تا اینجا بررسی شد اشکال بسیار ساده‌ای هستند که بوسیله روابط ریاضیشان توسط موتور تفسیر ، رسم می‌شوند ولی برای تولید اشکال سه‌بعدی پیچیده کافی نیستند. VRML گره‌های زیر را برای ایجاد اشکال دلخواه ارائه کرده است:

♦ **Coordinate3** : اولین گام در خلق یک شکل سه‌بعدی هندسی (بدون انحنای) ، تعیین مختصات تمام رئوس آن می‌باشد. این کار با گره Coordinate3 انجام می‌شود. به عنوان مثال فرض کنید که می‌خواهیم یک هرم بوجود بیاوریم. ابتدا باید مختصات پنج راس آن تعیین شود:

```
Coordinate3 {
    Point [
    -1 0 -1,
    1 0 -1,
    1 0 1,
    -1 0 -1,
    0 1 0
    ]
}
```



♦ **IndexFaceSet** : در دومین گام باید وجوه^۱ جسم ، تعریف شوند. اینکار با تعیین رئوسی که یک سطح از جسم را تشکیل می دهند ، توسط گره IndexFaceSet انجام می شود. ساختار این گره بصورت زیر است :

```
IndexFaceSet {
    CoordIndex [
        V0,V1,V3,...,Vn-1,
        V0,V1,V3,...,Vn-1,
        V0,V1,V3,...,Vn-1,
        V0,V1,V3,...,Vn-1,
    ]
}
```

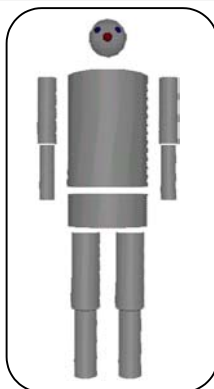
این گره با توجه به گره Coordinate3 ، "اندیس گره‌هایی" که یکی از وجوه شکل را مشخص می کنند ، تعیین می نماید. برای هرم بالا اندیس گره‌ها بصورت زیر خواهد بود :

```
IndexFaceSet {
    CoordIndex [
        0,4,1,-1,
        1,4,2,-1,
        2,4,3,-1,
        3,4,0,-1,
        0,1,2,3,-1
    ]
}
```

دقت کنید که ۱- علامت ختم یکی از وجوه و شروع وجه بعدی از شکل هندسی است. مثلاً قاعده هرم بالا ، در برگیرنده رئوس با اندیس (۰ ، ۱ ، ۲ ، ۳) است و ۱- پایان رئوس این وجه را اعلام می نماید. مجموعه اندیسها در درون علائم [] تعریف می شوند و هر یک از اندیسها با علامت و از هم جدا می شوند.

برای روشتر شدن تعریف اشیاء فوق در فایل VRML ، در ادامه با مثالی سعی شده است تا از طریق اشکال هندسی ساده مثل استوانه و کره ، ساختار نمادین زیر از بدن یک انسان مدل سازی شود:

^۱ Face



```
#VRML V1.0 ascii
# A simple humanoid body made up of cylinders and spheres
# Built by hand with a text editor!
```

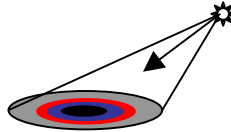
```
DEF Body Separator {
  DEF Pelvis Separator {
    Cylinder { radius 0.2 height 0.15 }
    DEF Chest Separator {
      Transform { translation 0 0.350 0 }
      Cylinder { radius 0.2 height 0.50 }
      DEF LeftUpperArm Separator {
        Transform { translation -0.275 0.10 0 }
        Cylinder { radius 0.05 height 0.30 }
        DEF LowerArm Separator {
          Transform { translation 0 -0.280 0 }
          Cylinder { radius 0.04 height 0.25 }
        }
      }
    }
    DEF RightUpperArm Separator {
      Transform { translation 0.275 0.10 0 }
      Cylinder { radius 0.05 height 0.30 }
      DEF LowerArm Separator {
        Transform { translation 0 -0.280 0 }
        Cylinder { radius 0.04 height 0.250 }
      }
    }
  }
  DEF Head Separator {
    Transform { translation 0 0.425 0 }
    DEF Skull Separator {
      Sphere { radius 0.1 }
    }
  }
}
```


◀ نورافکن نقطه‌ای^۱: این منبع نور، پرتوهای نوری را در تمام جهات به صحنه VR می‌تاباند (همانند یک لامپ معلق و آویزان):

```
PointLight {
On TRUE / FALSE          روشن یا خاموش بودن منبع نور
intensity x              شدت منبع نور (عددی بین صفر-خاموش- و ۱/۰-روشن-)
color r g b             رنگ منبع نور بر حسب سه مولفه قرمز/سبز/آبی
location x y z          مختصات موقعیت منبع نور
}
```

◀ نورافکن صحنه^۲: این منبع نور، پرتوهای نوری را در حالت مخروطی شکل به صحنه VR می‌تاباند تا بخشی از صحنه بطور خاص نورپردازی شود:

```
SpotLight {
On TRUE / FALSE          روشن یا خاموش بودن منبع نور
intensity x              شدت منبع نور (عددی بین صفر-خاموش- و ۱/۰-روشن-)
color r g b             رنگ منبع نور بر حسب سه مولفه قرمز/سبز/آبی
location x y z          مختصات موقعیت منبع نور
direction x y z         بردار جهت نور
dropOffRate dor         میزان تفرق نور از مرکز به سمت لبه‌های قاعده مخروط
cutOffAngle cof         زاویه‌ای که تحت آن زاویه، نور به صحنه تابانیده می‌شود.
}
```



◀ نورافکن مستقیم^۳: این منبع نور، پرتوهای نوری را به موازات یک بردار به صحنه VR می‌تاباند:

```
DirectionalLight {
On TRUE / FALSE          روشن یا خاموش بودن منبع نور
intensity x              شدت منبع نور (عددی بین صفر-خاموش- و ۱/۰-روشن-)
color r g b             رنگ منبع نور بر حسب سه مولفه قرمز/سبز/آبی
direction x y z         بردار جهت نور
}
```

^۱ Point Light
^۲ Spot Light
^۳ Directional

◀ برای رنگ‌آمیزی سطوح اشیاء در VRML می‌توان از گره `{ Material}` استفاده کرد:

```
Material {
  ambientColor r g b
  diffuseColor r g b
  specularColor r g b
  emissiveColor r g b
  shininess s
  transparency t
}
```

با گره `Material` علاوه بر رنگ سطوح، می‌توان ویژگی‌های درخشندگی، شفافیت و خصوصیات دیگر سطح اشکال را تعریف کرد. در گره `Material` می‌توان از فیلد `emissiveColor` برای نمایش اشیاء مشتعل، فیلد `specularColor` برای اشیاء با درخشش متالیک (فلزگونه) و از فیلد `ambientColor` برای اجسامی که باید کسری از نور صحنه را مثل یک جسم صیقلی منعکس کنند، استفاده کرد.

◀ `Fog`: گره‌ای برای کدر و مات کردن رنگ صحنه در محیط VR است.

◀ `Background`: با این گره می‌توان پس‌زمینه صحنه را طراحی و رنگ‌آمیزی کرد. مانند طراحی پس‌زمینه آسمان، زمین، جنگل و ...

یکی از قابلیت‌های مهمی که به VRML 2.0 اضافه شد، پشتیبانی از منابع صدا بود که با دو گره `Audioclip` و `Sound` در اختیار طراح قرار گرفت. گره `Sound` شبیه نورافکن صحنه عمل می‌کند یعنی صدا با استفاده از فیلد شدت صدا تحت کنترل طراح می‌باشد. این گره دارای فیلدهای مکان، بردار جهت و شدت صدا می‌باشد.

گره `URL Audioclip` محل قرار گرفتن فایل صدا را در شبکه اینترنت مشخص می‌کند. (در VRML 2.0 فایل‌های MIDI یا فایل‌های WAV پشتیبانی می‌شوند).

هدف از این فصل آشنایی با اصول کلی `HTTP`، `WEB`، `HTML` و `VRML` بوده است؛ مباحث و جزئیات هر یک از مقولات بررسی شده بسیار وسیع و پرحجم هستند و برای آشنایی دقیق و کاربردی باید به مراجع آنها مراجعه کرد.

(۷) مراجع این فصل

مجموعه مراجع زیر می‌توانند برای دست آوردن جزئیات دقیق و تحقیق جامع در مورد مفاهیم معرفی شده در این فصل مفید واقع شوند.

"Computer Networks" , Andrew S.Tanenbaum, Third Edition, Prentice-Hall, 1996.	
"Internet and Intranet Engineering" , Daniel Minoli , McGraw-Hill, NewYork 1997.	
"Internetworking with TCP/IP", Commer D.E. Prentice-Hall, 1996.	
"Special Edition Using VRML",Atephen Matsuba , Bernie Roehl, 1998.	
"CGI Programming Unleashed , Eugene Eric Kim , 1996 by Sams.net Publishing.	
"HTML 3.2 and CGI", Professional Reference Edition,UNLEASHED , John December and Mark Ginsburg, Sams.net Publishing.	
"CGI Manual of Style" , Robert McDaniel, ISBN: 1-56276-397-0.	
"Teach yourself VRML 2.0 in 21 days", Chris Arrin Bruse Campbell, 1997.	
RFC1945	Hypertext Transfer Protocol -- HTTP/1.0. T. Berners-Lee, R. Fielding & H. Frystyk. May 1996
RFC2068	Hypertext Transfer Protocol -- HTTP/1.1. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee. January 1997
Microsoft Internet	URL : http://www.microsoft.com/internet
The VRML Repository	URL : http://www.sdsc.edu/vrml/
The HTML Guru	URL : http://members.aol.com/htmlguru/index.html
Tim Berners-Lee's Style Guide	URL : http://www.w3.org/hypertext/WWW/Provider/Style/Overview.html
The Home of the WWW Consortium	URL : http://www.w3.org/
The World Wide Web FAQ	URL : http://www.boutell.com/faq/
The 24-Hour HTML Café	URL : http://www.mcp.com/sams/books/235-8/
The Developer's JumpStation	URL : http://oneworld.wa.com/htmldev/devpage/dev-page.html