# Web programazio plataformak zerbitzari ingurunean. LAMP aplikazioak

## Web programazioaren ezaugarriak

Ziur badakizula zer den web orrialde bat, eta baita bertara heltzeko jarraitu behar dituzun pausoak ere, nabigatzailean URL bat idatziz, hau pantailan kargatuko da eta ikusi ahal izango duzu.

Web orrialde bat zure ordenagailuan deskargatzen denean, bere edukiak erakutsi beharrekoa zehazten du. Lengoaia hau markatze lengoaia batean programatua egongo da, etiketekin sortua, eta HTML edo XHTML izan daiteke. Orrialdea osatzen duten etiketen osatzen duten atalen helburua deskribatzen dute. Beraz, testu bat izenburua den, taula baten atala den edo parrafo sinple bat den esango diguten etiketak daude.

Gainera, webgune abat ondo egituratua badago, orrialdea erakusteko erabili beharreko estiloa beste fitxategi batean egongo da gordeta. CSS fitxategian. Estilo orria zein den web orrian bertan esaten da eta batera deskargatuko dira biak webgunea ordenagailuan ikusi ahal izateko.

Beraz, bi fitxategi hauek zerbitzaritik ordenagailura deskargatuko dira eskari bati erantzuna ematen. Jarraitzen den prozesua honela laburbildu dezakegu:

1. Zure ordenagailuak zerbitzariari webgunea eskatzen dio, .html, .htm edo .xthml luzapenarekin.
2. Zerbitzariak bere orrialde biltegian eskatutako orria bilatzen du.
3. Zerbitzariak orria aurkitu eta hartuko du.
4. Azkenik nabigatzailera bidaliko da honek bere edukia erakutsi dezan.

Hau bezero-zerbitzari komunikazio arrunt bat da. Bezeroa da eskaria egiten duena eta komunikazioa hasten duena eta zerbitzaria da eskaria jaso eta berataz arduratzen dena. Gure kasuan nabigatzailea da web bezeroa.

## Web orrialde estatikoak eta dinamikoak (I)

Aurreko adibidean ikusitako web orrialdeak web orrialde estatikoak dira. Orrialde hauek bere bukaerako forman daude gordeta, sortu ziren bezala, bere edukia ez delako aldatzen. Erabilgarriak dira informazio konkretua erakusteko eta kargatzen diren bakoitzean informazio bera erakutsiko dute. Aldatuak izan daitezen programatzaileak bere edukia aldatu edo eguneratu beharko du.

Web orrialde estatikoei kontrajarrita, suposatuko duzuen moduan, web orrialde dinamikoak ditugu. Orrialde hauek, izenak dioen bezala, eduki ezberdina bistaratuko dute hainbat aldagairen arabera: erabiltzen ari zaren nabigatzailea, identifikatzeko erabili duzun erabiltzailea, edo aurretik burutu dituzun ekintzak.

Web aplikazio dinamikoen artean oso garrantzitsua da bi mota hauek ezberdintzea:

* Nabigatzailean exekutatzen den kodea dutenak. Orrialde hauetan kode exekutagarria, normalean JavaScript, HTML (edo XHTML) barruan kokatzen da eta orrialdearekin batera deskargatzen da. Nabigatzaileak orria pantailan bistaratzean, bertan datorren kodea exekutatuko du. Kode honek funtzionalitate ezberdinak. Kode honek funtzionalitate ezberdinak izan ditzake, animazioak erakustetik orrialdearen itxura eta edukia guztiz aldatzera.
* Dakizun bezala interneteko orri asko daude .html, .xhtml edo .htm luzapena ez dutenak. Orrialde hauetako askok .php, .asp, .cgi edo .aspx. Hauetan nabigatzaileak deskargatzen duen edukia webgune estatiko baten antzekoa da: HTML (edo XHTML). Aldatzen dena eduki hau lortzeko modua da. Orain arte ikusi dugunak ez bezala, orrialde hauek ez daude zerbitzarian gordeta; edo hobeto esanda, gordeta dagoen edukia z da nabigatzailera bidaltzen denaren berdina. Orrialde hauen HTML edukia programa bat exekutatuz lortzen dena da eta exekuzio hori web zerbitzarian gertatzen da.

Webgune dinamiko baten funtzionamenduaren eskema honakoa da:

1. Zure ordenagailuko web bezeroak (nabigatzailea) zerbitzariari web orrialdea eskatuko dio.
2. Zerbitzariak webgunea bilatu eta hartuko du.
3. Orrialde dinamiko bat baldin bada, hau da, HTML orrialdeko edukia eskuratzeko exekutatu beharra badauka, web zerbitzariak kodea exekutatzeaz arduratzen den moduluarekin kontaktatu eta kodea bidaliko dio.
4. Exekuzio prozesuaren barruan, informazioa eskuratu beharra gerta daiteke, adibidez datu-baseko erregistroak begiratuz.
5. Exekuzioaren emaitza HTML formatuan dagoen orrialde bat izango da, webgune estatiko baten oso antzekoa.
6. Web zerbitzariak erantzuna nabigatzaileari bidaliko dio, eta honek prozesatu eta pantailan bistaratuko du.

Prozedura hau behin eta berriz errepikatzen da webguneak kontsultatzen ditugunean. Adibidez, zure Gmail kontua kontsultatzen duzunean egiten duzun lehen gauza zure erabiltzaile izena eta pasahitza sartzea da. Ondoren, arruntena zerbitzariak zure sarrerako ontzia erakustea da, non zure kontuan jasotako mezuak bistaratzen diren. Pantaila hau orrialde dinamiko baten adibide argia da.

Argi dago nabigatzaileak ez diela orri berdina bidaltzen erabiltzaile guztiei, era dinamikoan sortuko du konektatzen den erabiltzailearen arabera. Orrialdea sortzeko erabiltzailearen datuak jasoko dituen programa bat exekutatuko du (kontaktuak, jasotako mezu zerrenda) eta beraiekin osatzen du zerbitzaritik jasoko duzun orrialdea.

## Web orrialde estatikoak eta dinamikoak (II)

Web orrialde dinamikoen erabilera webgune bat eraikitzeko aukera onena dela uste baduzu ere, beti ez da horrela. Argi dago potentzia eta malgutasun gehien duen aukera dela, baina webgune estatikoek ere badituzte abantailak:

* Ez da beharrezkoa programatzen jakitea webgune estatikoak bakarrik darabiltzan webgune bat egiteko. HTML eta CSS ezagutzak izatea nahikoa litzateke, eta hau ere ez da benetan beharrezkoa; webguneak diseinatzeko programa bat erabili dezake.
* Webgune estatikoen edukia ez da inoiz aldatzen, eta hau ere abantaila bat izan daiteke kasu askotan. Adibidez, orrialde baten helbidea gordetzea nahi baduzu eta webgune dinamiko bat baldin bada berriz bistaratzean edukia ezberdina izan daiteke. Edo zuk sortutako gune bat Google-en altan eman nahi duzunean.

Google-ek webgune bat bere bilaketa emaitzetan erakusteko lehenik eta behin bere edukia indexatu behar du. Hau da, programa batek webgunearen eduki dena pasatzen du edukia begiratzen eta sailkatzen. Webgunea era dinamikoan sortzen bada baliteke edukiaren atal bat ez izatea bilatzailearentzat ikusgarria eta ez da indexatuta geldituko. Orrialde estatikoak dituen webgune baten hau ez litzateke gertatuko.

Gainera, programaren exekuzioan zehar baliteke datu-base bat kontsultatu beharra izatea. Hau da, web orrialde dinamiko bat sortzeko hainbat baliabide eskatuko dizkigu zerbitzariaren aldean.

Baliabide hauek instalatu eta mantendu egin behar dira. Orrialde estatikoek zure nabigatzailearekin komunikatuko den zerbitzari bat bakarrik behar dute. Gainera, zuren ekipoan gordetako webgune estatiko bat ikusteko ez duzu zerbitzaririk behar. Fitxategi hauek edozein euskarritan gorde daitezke eta zuzenean nabigatzailean ireki.

Orrialde estatikoak erabiliz webgune bat sortzea nahi baldin baduzu, kontuan izan mugak dituela. Desabantaila nagusia aipatu dugu lehen: edukian aldaketak egin nahi izanez gero era manualean egin behar dira, zerbitzarian gordeta dagoen orrialdea aldatuz. Beraz mantentze lanak handiegiak izan daitezke.

Ze teknologia erabiliko zenuke webgune pertsonal bat sortzeko? Beharrezkoa litzateke orrialde dinamikoak erabiltzea? Ze mantentze eta eguneratze lan egin beharko zenituzke?

Interneten sortu ziren lehen web orrialdeak estatikoak izan ziren. Orrialde estatikoz osatutako web hau lehenengo belaunaldikoa dela esan dezakegu. Bigarren belaunaldia web dinamikoei esker sortu zen. Web dinamikoak oinarritzat hartuta teknologia berriak garatuz joan dira, eta hauei esker da Internet gaur egun den bezalakoa.

### Web aplikazioak

Web aplikazioek web orrialde dinamikoak darabiltzate, zerbitzari batean exekutatzen direnak eta nabigatzailean bistaratu. Ekintza ugari burutzeko web aplikazioak aurkituko dituzu. Agertu ziren lehenetarikoak posta elektronikoen bezeroak izan ziren, nabigatzaile baten bidez jasotako mezuak irakurri eta bidaltzea ahalbidetzen diten aplikazioak.

Gaur egun ekintza ezberdin ugaritarako web aplikazioak ditugu: testu editoreak, edota irudi editore edo biltegiak esate baterako. Aplikazio hauek aplikazio tradizionalekin alderatuz gero hainbat abantaila eta desabantaila dituzte.

#### Web aplikazioen abantailak

* Ez da beharrezkoa ekipoetan instalatzea. Gailu bakarrean instalatu eta exekutatzen dira, zerbitzarian, eta honekin nahikoa da hainbat ekipotan erabiltzeko aukera izateko.
* Ekipo bakarrean instalatuta daudenez erraza da kudeatzea (segurtasun kopiak egin, erroreak zuzendu, eguneratu...)
* Web nabigatzaile bat duen sistema orotan erabili daitezke, berdin da duen sistema eragilea edo dituen ezaugarriak.

#### Web aplikazioen desabantailak

* Aplikazioaren interfazea web nabigatzailean bistaratzen den orria da. Honek interfazearen ezaugarriak webgunearenera mugatzen ditu.
* Zerbitzarirako konexioaren menpe gaude erabili ahal izateko. Konexioak huts egiten badu ezingo dugu aplikaziora sartu.
* Nabigatzailean bistaratzen den informazioa zerbitzaritik dator. Hau dela eta aplikazio mota batzuk ez dira egokiak web aplikazio bezala garatzeko (multimedia edukia dutenak esaterako, bideoak editatzeko aplikazioak kasu).

Gaur egun web aplikazio askok darabiltzate web orrialde dinamikoek eskaintzen dituzten abantailak. Berauen edukiaren zati handi bat datu-base batean gordeta dago. Drupal, Joomla! Eta antzeko aplikazioek bi atal ezberdindu eskaintzen dituzte:

* Kanpoko atal bat edo front-end, erabiltzaile gehienak ikusten dituzten orrialdeen bilduma dira (kanpoko erabiltzaileak).
* Barneko atala edo back-end, edukia sortzen duten erabiltzen dituzten orrialde dinamikoak dira eta web aplikazioak administratzen ditu (barne erabiltzaileak) edukia sortzeko, antolatzeko, kanpo itxura erabakitzeko…

# Kodearen exekuzioa zerbitzarian eta bezeroan

Ikusi genuen bezala, zuren nabigatzaileak web zerbitzari bati orrialde bat eskatzen dionean, baliteke zuri bidali aurretik eduki hori lortzeko programa bat exekutatu beharra izatea. Programa hau da, hein batean edo osotasunean, zure webgunea sortzen duena. Kasu hauetan kodea zerbitzarian exekutatzen da.

Bestalde, web orrialde bat zure nabigatzailera iristen denean baliteke exekutatu beharreko programa edo kode atalen bat izatea. Kode hori, normalean Javascript, nabigatzailean exekutatuko da eta gainera, orrialdearen edukia aldatu dezake edota testuei edota objektuei animazioa gehitu edo ta egiaztapenak egin formulario batean.

Bi teknologia hauek elkarren osagarriak dira. Honela, web aplikazioaren kasura itzulita, zure mezuak eta edukia datu-basetik eskuratzeaz arduratzen den programa zerbitzarian exekutatzen da, nabigatzaileak mezu bat idatzi eta nori bidaltzea nahi duzun ipintzea ahaztu duzula esaten dizun mezua sortzen duen atala exekutatzen du.

Bereizketa hau egiteko arrazoia hasieran bezeroan exekutatzen zen kodeak zerbitzariko ataleko datuetara sarrerarik ez izatea da. Irtenbidea webgunea zerbitzarian sortu eta nabigatzailera bidaltzea zen. Nahiez eta gaur egun AJAX bezala ezagutzen den teknika dagoen, bezeroan exekutatu Javascript kodeari informazioa eskuratzeko aukera ematen diona.

Gure adibidearekin jarraituz, irakurri nahi duzun mezu baten gainean klik eginez gero Javascript kodeak akzioa antzeman eta Internet bidez kontsultatu dezake mezu hori bera. Hau guztia orrialdetik bertatik irten gabe orrialdearen edukia aldatzeko zerbitzaritik lortutako informazioarekin.

Modulu honetan zerbitzarian exekutatuko diren aplikazioak sortzen ikasiko duzu. Beste modulu batean bezeroan exekutatzen diren aplikazioak sortzen ikasiko duzun bitartean.

# Zerbitzari inguruneko programazio teknologiak

Aplikazio bat programatzen duzunean programazio lengoaia bat erabiltzen duzu. Adibidez, Java lengoaia erabiliko duzu sistema eragile desberdinetan exekutatuko den programa bat egiteko. Aplikazio bakoitza programatzean garapen ingurune bat eta kode liburutegiak bezalako erremintak erabiliko dituzu. Gainera, behin garapena bukatutakoan aplikazio honek hainbat osagai beharko ditu bere exekuziorako.

Bloke honetan web zerbitzari batean exekutatu daitezkeen aplikazioak programatzeko teknologia ezberdinak ikasiko dituzu, eta baita nola erlazionatzen diren ere. Bakoitza erabiltzearen abantailak eta desabantailak ere ikusiko dituzu.

Zerbitzari batean aplikazio bat exekutatzeko beharko dituzun osagai nagusiak hauek dira:

* Web zerbitzari bat, web bezeroen eskariak jaso (normalean nabigatzaileenak) ea eskatutako web orrialdea bidaliko dutenak (behin sortu duenean). Web zerbitzariak web orrialdea sortzeko jarraitu beharreko prozedura jakin beharko du: ze moduluk exekutatuko duen kodea, eta nola komunikatu behar den berarekin.
* Kodea edo programa exekutatzeaz eta web orrialdea sortzeaz arduratuko den modulua. Modulu hau nolabait web zerbitzariarekin integratu behar da, eta erabiltzen ari garen lengoaiaren eta teknologiaren araberakoa da izango da.
* Datu-base aplikazio bat, normalean zerbitzari bat ere izango dena. Modulu hau ez da nahitaezkoa baina praktikan datu kopuru handia darabilten aplikazio guztiek erabiltzen dute.
* Aplikazioak garatzeko erabiltzen ari zaren programazio lengoaia.

Erabili beharreko osagaiez gain aplikazioaren kodea nola antolatuko duzun esatea ere garrantzitsua da. Web aplikazioen garapenean erabiltzen diren arkitektura askok kodea geruzatan edo mailatan banatzen dute.

Diseinua geruzatan banatzearen arrazoia funtzio logikoak banatuta egotea da, bakoitza zerbitzari ezberdin batean exekutatu ahal izateko, beharrezkoa baldin bada.

Aplikazio batean, era orokorrean aurkezpen funtzioak (bukaerako erabiltzaileari aurkeztuko zaizkion datuei forma emateaz arduratzen da), logikoak (datuak erabiltzen ditu prozesu bat exekutatzeko eta emaitza bat lortzeko) eta sarbidekoak(biltegiratze eremuan datuak sartu eta bertatik datuak eskuratzeko).

Geruza bakoitza aurreko funtzio batez edo bat baino gehiagoz arduratu daiteke. Adibidez, hiru geruzatako aplikaziotan honakoa aurkitzen dugu:

* Bezero geruza, erabiltzaile interfazearekin zerikusia duen dena programatzeko erabiltzeko, erabiltzaileak ikusiko duen eta erabiliko duen atala.
* Tarteko geruza, aplikazioaren funtzionalitatea programatuko duzu bertan.
* Datu-sarbideko geruza, aplikazioaren informazioa datu-base batean gordetzeaz eta datuak eskuratzeaz arduratuko dena.

# Arkitekturak eta plataformak

Aplikazio bat programatzen hasi aurretik hartu beharko duzun lehen erabakia erabiliko duzun arkitektura aukeratzea izango da. Gaur egun aukera ezberdinak dituzu aukeratzeko:

* JavaEE (Enterprise Edidion), lehen J2EE bezala ezagutzen zena. Java lengoaian aplikazioak sortzeko plataforma da. Datu-base kudeatzaile ezberdinekin lan egin dezake, eta aplikazioak era modularrean programatzeko hainbat libreria eta espezifikazio eskaintzen ditu.

Sun eta Oracle bezalako enpresa handien babesa du. Tamaina ertain eta handiko aplikazioentzat irtenbide egokia da. Duen abantaila handienetako bat merkatuan dauden libreria kopuru handia da, eta lengoaia ezagutzen duten programatzaile kopuru handia ere bai.

Arkitektura honen barruan teknologia ezberdinak daude, JSP orriak eta servletak esate baterako, biak webgune dinamikoen sorrerara bideratuak. Edota EJBak, normalean web aplikazioari logika ematen diotenak.

* AMP. Apache, MySQL eta PHP/Perl/Python siglak dira. Zerbitzarirako erabiltzen den sistema eragilearen arabera LAMP (Linux), WAMP (Windows) edo MAMP (Mac) aukerak ditugu. Beste osagai batzuk erabiltzea ere posible da, PostgreSQL datu-base kudeatzailea esate baterako.

Arkitektura honen osagai guztiak kode irekikoak dira (open source). Ikasketa prozesu sinple bat izanda, tamaina txiki edo ertaineko aplikazioak garatzen lagunduko digun plataforma bat da. Abantaila bat atzen duen komunitatea da, eta eskuragarri dauden kode askeko aplikazio kopuru handia.

Badaude instalazio bakarrean AMP plataforma osoa duten software paketeak. Batzuk instalatzeko beharrik ere ez dute, eta sistema eragile ezberdinetarako dago eskuragarri (Linux, Windows edo MAC). Ezagunenetako bat XAMPP da.

<http://www.apachefriends.org/es/xampp.html>

* CGI/Perl. Bi osagaien arteko konbinazioa da. Perl, kode libreko lengoaia potente bat, eta CGI, web zerbitzariari programa generikoak exekutatzen uzten dion estandar bat. Programa generiko hauek edozein lengoaiatan idatziak egon daitezke (baita C edo Python ere) eta exekuzioaren emaitza bezala web orrialde bat (HTML) itzuliko dute.

Konbinazio honen desabantaila handiena CGI motela dela da, baina badaude azkartzeko metodoak. Bestalde, Perl lengoaia oso potentea da eta erabiltzaile komunitate handi bat du.

* ASP.Net Microsoftek proposatutako arkitektura da aplikazioak garatzeko. .Net plataformaren barruan web orrialde dinamikoak garatzeko atala da.

Programazio lengoaia Visual Basic.Net edo C# izan daiteke. Arkitekturak, ISS, Microsoften web zerbitzaria darabil eta datu-base kudeatzaile ezberdinetatik hartu dezake informazioa, horien artean Microsoft SQL Server.

.Net arkitekturaren abantaila nagusietako bat aplikazioen garapen eta zabalkuntzarako beharrezkoak diren erreminta guztiak eskaintzen dituela da. Adibidez, bere garapen ingurune propioa dauka, Visual Studio, baina beste aukera batzuk ere badaude eskuragarri. Desabantaila handiena jabetzako kodea duen plataforma komertziala dela da.

## Web programazioko arkitektura baten aukeraketa

Ikusi duzun bezala web bat garatzen hasi aurretik hainbat erabaki hartu behar dituzu. Erabiliko duzun arkitektura, programazio lengoaia, garapen ingurunea, datu-base kudeatzailea, web zerbitzaria eta baita zure aplikazioa nola egituratuko duzun ere.

Erabaki zuzena hartzeko ondorengo galderetan pentsatu beharko duzu:

* Ze tamaina dauka proiektuak?
* Ze programazio lengoaia ezagutzen ditut nik? Merezi du lengoaia berri bat ikasteko esfortzua egiteak?
* Kode irekiko erremintak erabiliko ditut edo jabetzakoak? Zein da irtenbide komertzialak erabiltzearen kostua?
* Aplikazioa nik bakarrik programatuko dut edo programatzaile talde batean egongo naiz?
* Badaukat jadanik web zerbitzari bat edo datu-base zerbitzari bat edo nahi dudana aukeratzeko askatasuna duk?
* Ze lizentzia ipiniko diot garatuko dudan aplikazioari?

Galdera hauei emandako erantzuna aztertuz zure beharretara gehien egokitzen den arkitekturak zeintzuk diren ikusiko duzu.

**Proiektu bakoitzak bere berezitasunak ditu. Nahiz eta arkitektura batzuek besteak baino potenteagoak diren, ez dago kasu guztietan besteak baino hobea denik. Aplikazio bakoitzerako garapen arkitektura aldatzeak pena merezi duela uste duzu? Beharrezkoa da kasu guztietan?**

## Web zerbitzariarekin integrazioa

Dakizun bezala zerbitzari eta bezero baten arteko komunikazioa HTTP protokoloari esker egiten da. Web aplikazioen kasuan HTTP erabiltzailearen eta aplikazioaren arteko lotura da. Erabiltzaileak egiten duen informazio eskaera oro HTTP eskaera bidez transmititzen da eta erantzuna ere HTTP bidez eskuratzen da.

Zerbitzariaren aldean eskari hauek web zerbitzariak prozesatzen ditu. Berak esango du eskari hauek nola prozesatu. Ikusi dugun arkitekturetako bakoitzak web zerbitzariarekin integratzeko modu bat dauka aplikazioaren kodea exekutatzeko.

### CGI

Teknologiarik zaharrena CGI da. CGI plataforma askotan existitzen den protokolo estandar bat da, eta web zerbitzari gehienak inplementatzen dute. Web zerbitzari batek kanpoko programa batek web orrialdea sortzea delegatzeko zer egin behar duen esaten du. Programa horiek CGI gidoiak bezala ezagutzen dira, programatzeko erabili den lengoaia dena delakoa delarik (nahiz eta Perl bezalako gidoi lengoaietan programatzen diren).

CGIren arazo nagusia honakoa da, CGI gidoi bat exekutatzen denean sistema eragileak prozesu berri bat sortu behar duela. Honek baliabideen kontsumo handiagoa dakar eta exekuzio denbora txikiagoa. Exekuzioa azkartzen duten irtenbide batzuk existitzen dira, FastCGI esaterako, eta baita web zerbitzari ingurunean gidoiak exekutatzeko beste metodo batzuk ere, Apachen Perl gidoiak exekutatzeko erabiltzen den **mod\_perl** adibidez.

### AMP

Bestalde, CGI erabiliz PHP lengoaian idatzitako kodea exekutatu daitekeen arren, PHP interpreteek ez dute metodo hau erabiltzen. **mod\_perl** modulua bezala, Apache web zerbitzarian instalatu daitezkeen modulu gehiago daude web orrialde dinamikoak exekutatzeko. **mod\_php** modulua da AMP plataformetan PHP gidoiak exekutatzeko erabiltzen dena, eta bere Pythonerako baliokidea **mod\_python** da.

Java EE arkitektura konplexuagoa da. Java EE aplikazioak zerbitzari batean exekutatzeko bi aukera ditugu: aplikazio zerbitzariak (JavaEE-ko teknologia guztiak inplementatzen dituztelako) eta servlet edukiontziak (espezifikazioaren atal bat bakarrik dutenak). Gure aplikazioaren tamainaren arabera eta erabilitako teknologien arabera, soluzio bat edo bestea instalatu beharko dugu.

### JavaEE

Java EE aplikazioentzat Sun-ek ziurtatutako aplikazio zerbitzarien zerrenda ikusi nahi izanez gero [honako](https://es.wikipedia.org/wiki/Java_EE#Servidores_de_Aplicaciones_Java_EE_5_certificados) webgunean dituzu.

Hala ere, kasu gehienetan ez da beharrezkoa aplikazio zerbitzari oso bat instalatzea, nahikoa izango da servlet edukiontzi batekin. Esparru honetan Tomcat nabarmendu behar dugu, servlet edukiontzien inplementazio esanguratsuena eta gainera kode irekikoa da. Aukeratutako soluzioa instalatu dugunean, web zerbitzariarekin integratu beharko dugu, servlet eta JSP orrialdetara zuzendutako eskariak bideratu ditzan.

Beste aukera bat orrialde estatiko eta dinamikoentzat soluzio bakarra erabiltzea da. Esate baterako, Tomcat-ek Apache ordeztuko duen HTTP zerbitzari propio bat dauka.

### ASP.Net

ASP.Net arkitekturak Microsoften ISS zerbitzaria darabil, ASP eta ASP.Net orrialde dinamikoen eskariak kudeatuko dituen moduluen euskarria daukana. Web zerbitzariaren administrazioak bertan instalatutako aplikazioen administrazioa du barneratuta.

# Lengoaiak

Webguneak programatzeko erabiltzen diren lengoaia batzuen eta besteen artean dauden ezberdintasun nagusiak hauek zerbitzarian exekutatzeko duten moduarekin dute zerikusia. Hiru talde handi bereiztu daitezke:

* **Gidoi lengoaiak (scripting):** programak zuzenean bere iturburu kode originaletik exekutatzen dira. Normalean testu lauan idatzitako fitxategi batean gordetzen dira. Web zerbitzariak gidoi lengoaian programatutako kodea exekutatzea behar duenean, eskaria interprete bati pasatuko dio eta honek programaren lerroak prozesatuko ditu eta web orrialdea sortu. Talde honetan ditugu: Perl, Python, PHP eta ASP (ASP.Net-en aurrekaria).
* **Kode natiboan konpilatutako lengoaiak (makina lengoaian):** iturburu kodea kode bitarrera itzulia izaten da exekutatu aurretik, prozesadorearen arabera. Web zerbitzariak programak era bitarrean gordetzen ditu eta deituak izatean zuzenean exekutatzen ditu.

Programa bitarrak web zerbitzaritik exekutatzeko modu nagusia CGI da. CGI erabiliz helburu orokorreko lengoaia batean programatutako kodea exekutatu dezake (adibidez C-n).

* **Bitarteko kodean konpilatutako lengoaiak.** Iturburu kodea prozesadorearekiko independentea den bitarteko kode batera itzultzen da, exekutatua izan aurretik. Javan programatutako aplikazioak honela exekutatzen dira eta horregatik exekutatu daitezke edozein plataformatan. Honela lan egiten duten Java EE (JSP orriak eta servletak) eta ASP.Net-ek.

ASP.Net eta Java EE inplementazio askotan JIT konpilazio prozedura bat erabiltzen da. Termino honek bitarteko kodea kode bitar bihurtzen den moduari egiten dio erreferentzia. Exekuzioa azkartzeko, konpilatzaileak deia jasotzean dena kode natibora itzuli dezake edo atal bat bakarrik. Lortutako kode bitarra beharrezkoa denean erabiltzeko gorde daiteke.

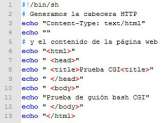
Aurrean aipatutako kodea exekutatzeko modu bakoitzak bere abantaila eta desabantailak ditu:

* Scripting lengoaien abantaila exekutatuak izateko itzulpen beharrik ez izatea da, eta honek eramangarritasuna hobetzen du. Programa bati aldaketak egin behar bazaizkio momentuan egin daitezke. Baina interpretazioaren errendimendua beste aukerena baino kaxkarragoa da.
* Kode natiboan konpilatutako lengoaiak dira exekuzioan azkarrenak, baina arazoak izaten dituzte web zerbitzariarekin integratzean. Helburu orokorreko programak dira eta ez daude pentsatuta zerbitzari ingurunean exekutatzeko. Adibidez, prozesuak ez dira berrerabiltzen eskariei erantzuteko: web zerbitzariari egiten zaion eskari bakoitzeko prozesu berri bat exekutatzen da. Gainera programak ez dira eramangarriak plataforma ezberdinen artean.
* Tarteko kodean konpilatutako lengoaiek aurreko bi aukeren arteko oreka bat eskaintzen dute. Bere errendimendua oso ona da eta plataforma ezberdinen artean eramangarria da arkitektura inplementatua baldin badago (JavaEE zerbitzari bat edo servlet edukiontzi bat).

## Markatze lengoaian kapsulatutako kodea

Webak orrialde estatikoetatik dinamikoetara eboluzionatzen hasi zenean erabili zen lehen

teknologietako bat kodea CGI bidez exekutatzea izan zen. CGI scriptak estandarrak dira, sistema eragilean exekutatzen dira eta irteera bezala webgune baten HTML lengoaia sortzen duten. Hori dela eta CGI scriptek web orrialdeak sortzeko sententziak dituzte kodean tartekatuta.



Adibidez, Linux sententzia bidezko CGI bat erabiliz webgune bat sortu nahi baduzu irudian bezalako zerbait egin beharko duzu.

Hau da webgune dinamikoak sortzeko modu nagusietako bat: HTML etiketak programen kodean gehitzea. Hau da, irudiko programen antzera kodeak HTML kodea idatziko duten sententziak izango ditu (kasu honetan echo sententziarekin). Era honetan programatzen dira CGI scriptak eta servletak.

Beste ikuspegi ezberdin bat programaren kodea HTML etiketen artean sartzea da. Era honetan web orrian aldatzen ez den HTML atala zuzenean idatzi daiteke eta programazio lengoaia era dinamikoan aldatuko den guztiarentzat erabiliko da.

Adibidez, HTML kode baten barruan PHP kodea gehitu daiteke. Adibidez:



Programazio metodologia hau ASP, PHP lengoaietan eta JavaEE-ren JSP orrietan erabiltzen da. Aipatu behar da Java EE servletak konpilatutako programak diren bitartean, JSP orrialdeak Java lengoaia HTML barruan kapsulatuta dutela eta web zerbitzarian era indibidualean gordetzen direla. JSP orri bat behar den lehen aldian hau servlet bilakatzen da eta orrialde berdinak jasotzen dituen ondorengo deietan berriro erabiliko da.

ASP.Net arkitekturan orrialde bakoitza bi fitxategitan banatzen da: HTML etiketak alde batetik eta erabilitako programazio lengoaia bat bestetik. Era honetan independentzia maila bat lortzen da aplikazioaren itxuraren eta eduki dinamikoaren kudeaketaren artean. Fitxategi horietatik abiatuz tarteko kode bat lortzen da eta hau da zerbitzariak gordetzen duena.