DigitAll

Formació en Competències Digitals

3

Creació de continguts digitals

DigitAll

Formació en competències digitals

Creació de contingut digital

Nivell C1

Creació de continguts digitals

ÍNDEX

DigitAll

Creació de
continguts digitals

3.1
DESENVOLUPAMENT DE CONTINGUTS

Creació de
continguts digitals

Nivell C1 3.1 Desenvolupament de continguts

Generació automàtica de text a partir de tècniques d'IA. GPT un cas pràctic

Generació automàtica de text a partir de tècniques d'IA. GPT un cas pràctic

Introducció

La generació automàtica de text ha experimentat avanços significatius gràcies al desenvolupament de tècniques d'intel·ligència artificial (IA). Aquestes tècniques permeten a les màquines comprendre i produir text de manera coherent i contextualment rellevant, en alguns casos, de manera molt similar a com ho faria un humà. En aquest text, explorarem algunes de les tècniques que s'han emprat per a desenvolupar aquestes IA i posarem alguns exemples amb GPT, probablement la IA més empleada per a generar text.

Models d'IA per a la generació automàtica de text

Existeixen diferents aproximacions a l'hora d'aconseguir que una IA generi text, lògicament, cadascuna proposa l'ús d'una sèrie d'eines i produeix un text d'unes determinades característiques. A continuació, enumerarem algunes i després passarem a parlar més específicament del model GPT.

- **Models basats en regles:** aquests models empren algorismes predefinits i regles específiques per generar text. Encara que poden produir resultats acceptables en escenaris simples, la seva capacitat per adaptar-se a contextos més complexos és limitada. Un tipus de tecnologia basada en regles és *ChatScript*, que es va dissenyar inicialment per crear bots.
- **Models de llenguatge estadístics:** aquests models empren tècniques estadístiques per analitzar patrons i estructures en grans conjunts de dades textuals. A través de l'aprenentatge automàtic, aquests models generen text basant-se en la probabilitat d'ocurrència de paraules i seqüències. *Spacy*, una llibreria de programari per al processament de llenguatges naturals, pertany a aquesta mena de models.
- **Xarxes neuronals recurrents (RNN, en anglès):** els RNN són models de IA que tenen en compte la seqüència i el context del text. Aquestes xarxes neuronals processen la informació

de manera seqüencial, la qual cosa els permet capturar dependències a llarg termini i generar text coherent i fluïu. Per exemple, *TensorFlow* ofereix una àmplia gamma de funcionalitats per a l'entrenament i desplegament de models de RNN, la qual cosa facilita la generació de text seqüencial i contextualment coherent.

Què és GPT-4?

GPT-4 *(Generative Pre-trained Transformer 4*) és una de les darreres incorporacions en el camp de la generació automàtica de text. Aquest model es basa en l'arquitectura Transformer, que ha demostrat la seva eficàcia en una àmplia gamma de tasques de processament del llenguatge natural.

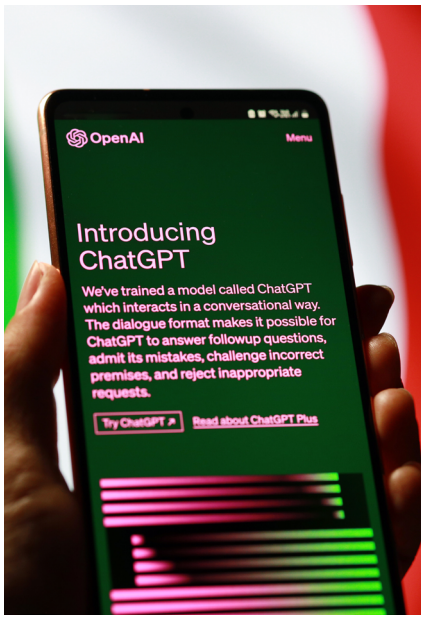
GPT-4 utilitza tècniques d'aprenentatge automàtic per predir la següent paraula o frase en funció del context proporcionat. A través d'un entrenament intensiu en grans conjunts de dades, GPT-4 ha adquirit un coneixement profund del llenguatge i pot generar text coherent i rellevant en diversos dominis. A diferència dels *RNN*, els models basats en Transformer utilitzen una arquitectura que permet capturar relacions a llarg termini en el text sense dependre exclusivament de l'ordre de les paraules.

Dos aclariments importants a l'hora d'emprar GPT-4 de manera efectiva. D'una banda, és important destacar que és necessari proporcionar al sistema una sèrie d'instruccions, que estudiarem en el següent nivell, que ens permetran optimitzar el resultat. D'altra banda, no hem d'oblidar que aquest sistema té algunes limitacions si hi accedim des de *ChatGPT*, la més destacada és que els basis de dades que es van emprar per al seu entrenament són anteriors a 2021, per la qual cosa el seu coneixement del món és limitat a partir d'aquest any. No obstant això, en la seva versió de pagament és possible incorporar connectors amb connectivitat a Internet o també podem recórrer al xat de *Bing*, que està connectat a Internet.

Què es pot fer amb GPT-4?

Aquesta eina ens proporcionarà una infinitat de possibilitats, fins i tot si només l'empram com a generador de text. El més senzill és demanar-li que ens expliqui qualsevol cosa: un fet històric, com funciona un dispositiu tecnològic, com es prepara una recepta, etc. Però també podem obtenir textos més complexos com a cançons o poemes.

En relació amb els temes que hem treballat en aquesta competència, GPT-4 ens pot ajudar a generar molt de contingut multimèdia, per exemple, li podem demanar que ens indiqui com aplicar una màscara de capa en una imatge, com aplicar efectes sobre un text o com generar una web usant *WordPress*.



Saber-ne més

Plataformes com Coursera, Udemy i edX ofereixen una àmplia varietat de cursos relacionats amb la intel·ligència artificial, el processament del llenguatge natural i la generació de text. Alguns dels cursos amb més èxit són "*Natural Language Processing*" de la Universitat de Stanford en Coursera o "*Deep Learning Specialization*" també a Coursera.

Nivell C1 3.1 Desenvolupament de continguts

Creació i edició de rutes al disseny gràfic

Creació i edició de rutes al disseny gràfic

La creació i edició de rutes és una de les eines més potents i versàtils en el camp del disseny gràfic. Encara que pot semblar complexa a primera vista, amb una comprensió clara de les seves capacitats i com s'apliquen, pot obrir un món de possibilitats per al dissenyador gràfic.

Què és una ruta de disseny gràfic i quina és la seva utilitat?

Les rutes, també conegudes com a "paths" en anglès, és una eina fonamental en el disseny gràfic digital. Són traços vectorials que es poden manipular per crear formes, línies i corbes complexes. Les rutes són molt versàtils i poden ser utilitzades tant per a definir zones de selecció sobre les quals aplicar efectes i canvis, com per al dibuix de formes.

Les rutes són essencials perquè defineixen exactament quina part d'una imatge se selecciona per aplicar un canvi de color, un efecte de desenfocament, entre altres. A més, les formes basades en rutes tenen l'avantatge de ser totalment editables i escalables sense perdre qualitat, la qual cosa les converteix en una eina valuosa per al disseny de logotips, icones, i qualsevol element gràfic que pugui necessitar ajustaments de grandària.

Creació i edició

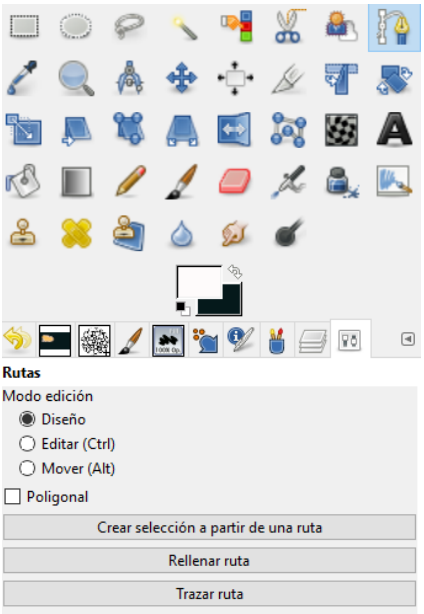
Emprarem el programari GIMP per explicar pas a pas la creació i edició d'una ruta de manera Inicial.

1 | Obre GIMP i la teva imatge

Obre GIMP i carrega la imatge amb la qual vols treballar. Pots fer això fent clic en "Arxiu" en la barra de menú superior, després selecciona "Obrir" i cerca la imatge que vols emprar.

2 | Obre l'eina de rutes

Fes clic en l'eina de rutes en el panell d'eines de GIMP. Aquesta es representa amb una icona que sembla un bolígraf/ploma del qual surt un traç amb vectors. Si no trobes la caixa d'eines, pots activar-la anant a "Finestres" en el menú, després selecciona "Eines recentment tancades" i finalment "Eines".



3 | Crea la teva ruta

Fes clic en el lloc on t'agradaria que comencés la teva ruta. Això crearà el primer node o ancora. Després, fes clic en el lloc on t'agradaria que acabés la teva línia o corba. Això crearà un segon node. Si vols crear una línia recta, pots deixar-ho així. Però si vols una corba, fes clic amb el botó esquerre del ratolí i arrossega cap al segon node. Veuràs aparèixer dues línies de color groc des del node que pots arrossegar per ajustar la corba.

4 | Crea formes més complexes

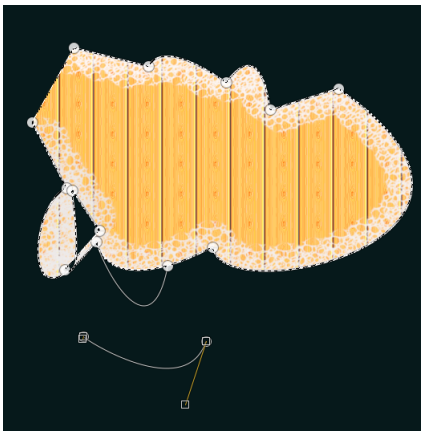
Per a crear formes més complexes, simplement continua fent clic per agregar més nodes. Pots fer clic al primer node quan hagis acabat per tancar la ruta.

5 | Edita la teva ruta

Pots ajustar la ruta en qualsevol moment fent clic i arrossegant qualsevol node. També pots fer clic i arrossegar les línies entre els nodes per ajustar les corbes, moure-la, emplenar-la amb color sòlid o patró i posar-li un traç o no.

6 | Aplica la teva ruta

Una vegada que estiguis satisfet amb la teva ruta, pots emprar-la per a diverses coses, com fer una selecció. Per a fer-ho, ves al panell de rutes (si no està visible, pots trobar-ho en "Finestres" > "Eines recentment tancades" > "Rutes"), fes clic dret en la ruta que acabes de crear i selecciona "De ruta a selecció".



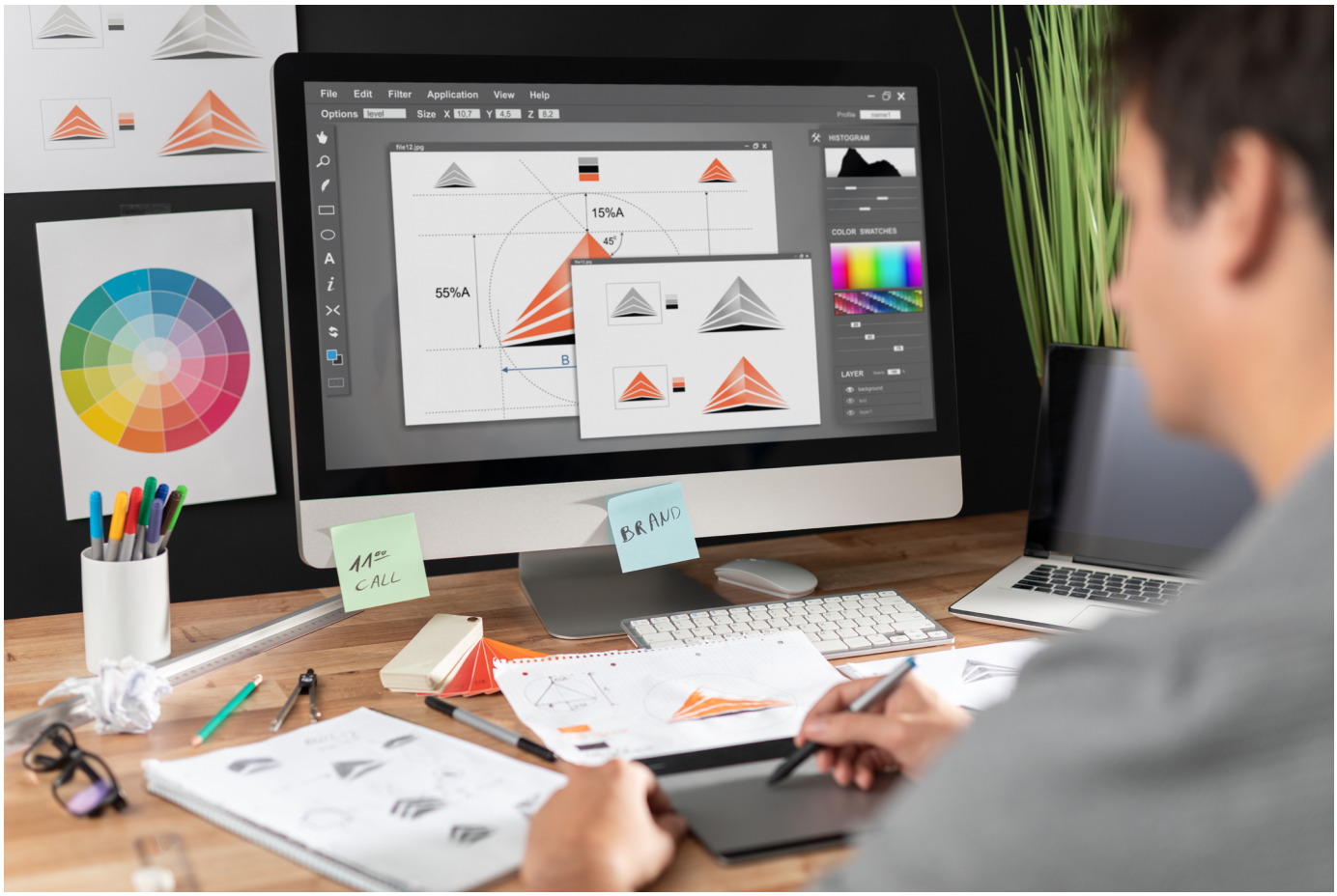
Aquest ha estat un cop d'ull general a la creació i edició de rutes en disseny gràfic. En nivells superiors, aprofundirem sobre

com maximitzar l'ús d'aquesta eina, amb exemples pràctics i tècniques més detallades.

Saber-ne més

L'eina de rutes té una corba d'aprenentatge lenta, però una vegada que es domina, pot ser increïblement útil. Al principi, pot ser útil treballar amb rutes simples i pràctiques de dibuix bàsiques per acostumar-se a com es comporten les rutes. Gradualment, pot experimentar amb formes més complexes i seleccions més detallades.

e.digitall.org.es/ruta-gimp

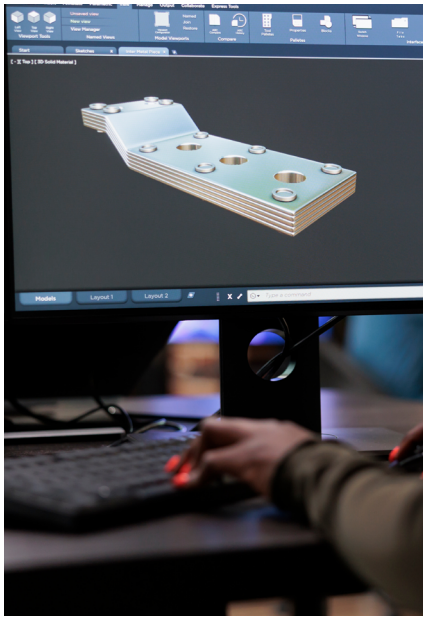


Creació de continguts digitals

Nivell C1 3.1 Desenvolupament de continguts

Eines per al disseny 3D: lliures i privatives

Eines per al disseny 3D: lliures i privatives



Introducció

El disseny 3D ha revolucionat la manera en què conceptualitzam i cream objectes virtuals. Gràcies a les capacitats avançades de modelatge, animació i renderitzat, el disseny 3D s'ha convertit en una eina essencial en disciplines com l'arquitectura, el disseny industrial, la medicina, l'enginyeria i l'art digital. En aquest text, explorarem la importància del disseny 3D i les eines disponibles, tant lliures com privatives, que permeten materialitzar les nostres idees en el món virtual.

Eines per al disseny 3D

Programari CAD

Veurem a continuació que moltes de les eines que esmentarem inclouen les lletres CAD en el seu nom. Quan parlem de programari CAD (de l'anglès *Computer-Aided Design*, en català Disseny Assistit per Ordinador) ens referim a una categoria de programari especialitzat que proporciona les eines per crear, modificar i optimitzar dissenys en format digital.

Així, el programari CAD permet als dissenyadors crear models virtuals en 2D o 3D d'objectes o sistemes. D'aquesta manera, utilitzant eines i funcions específiques, els usuaris poden dibuixar geometria precisa, aplicar restriccions i dimensions, realitzar simulacions o analitzar propietats físiques dels dissenys.

El programari CAD, com els programes que esmentarem a continuació, ofereix una sèrie d'avantatges sobre els mètodes de disseny tradicionals. D'una banda, en treballar en un entorn digital, els dissenyadors poden realitzar modificacions ràpides i precises en els dissenys, explorar diferents opcions i avaluar la seva viabilitat abans de la producció física. D'altra banda, atès que aquesta tecnologia substitueix el dibuix manual per un procés automatitzat, els projectes desenvolupats amb aquests programes es duen a terme en menys temps del que necessitaria un disseny clàssic.

Eines lliures

El programari lliure, com en tants altres camps, ha democratitzat l'accés al disseny 3D en ser gratuït i de codi obert. Probablement, el programari lliure de modelatge i animació 3D més emprat és Blender. De fet, és tan potent que és el preferit de molts professionals.

Encara que pot resultar una mica intimidant al principi, Blender és bastant senzill d'emprar i té una interfície intuïtiva. La seva popularitat es deu a diversos avantatges, d'una banda, és un programari multiplataforma que ens permetrà controlar tot el procés de creació (modelatge, animació, simulació, renderitzat, etc.). D'altra banda, la seva àmplia comunitat d'usuaris contribueix amb actualitzacions constants, recursos addicionals i tutorials en línia que poden ajudar-te a aprendre a emprar Blender.

No obstant això, a més de Blender, existeixen altres eines lliures per fer disseny 3D. Algunes de les més populars són:

- **FreeCAD:** és un modelador 3D paramètric de codi obert multiplataforma fet principalment per dissenyar objectes de la vida real de qualsevol grandària.
- **OpenSCAD:** és un programa CAD de codi obert que crea models 3D a partir de scripts o guions. Una vegada ens hem familiaritzat amb el llenguatge de programació que usa, a partir d'aquests guions podrem crear models 3D complexos.
- **3D Builder:** és una aplicació desenvolupada per Microsoft adequada per a usuaris sense experiència que permet veure, crear i personalitzar objectes 3D
- **Tinkercad:** és una eina gratuïta en línia per crear dissenys en 3D. És fàcil d'emprar i no requereix experiència prèvia en disseny 3D. De fet, s'usa en educació, ja que es basa a combinar formes simples per generar models complexos, com si es tractés d'un LLEC.

NOTA

La pel·lícula d'animació Next Gen, de Netflix (2018), va ser creada amb Blender íntegrament.

Eines privatives

Les eines privatives solen oferir major quantitat d'eines i característiques avançades que els programes lliures, però sobretot, un suport tècnic especialitzat per resoldre els problemes de l'usuari. A més, en general, solen tenir una interfície més amigable.

A continuació, es descriuran alguns dels programes de disseny 3D privatius més populars:

- **Autodesk 3ds Max:** és un programari àmpliament utilitzat en la indústria del disseny i la visualització 3D, ja que les seves versions originals daten dels noranta. Ofereix una àmplia gamma d'eines per a modelatge, animació, simulació i renderitzat. És popular en la indústria de l'entreteniment i s'ha emprat en pel·lícules com Blade: Trinity, Missió Impossible i Hellboy.
- **Autodesk Maya:** és un altre programa desenvolupat per Autodesk i és molt utilitzat en la indústria del cinema i l'animació. És conegut per la seva capacitat per a crear personatges 3D realistes i efectes especials impressionants. De fet, s'ha emprat en pel·lícules tan populars com Avatar o algunes seqüeles de la saga The Matrix.
- **Cinema 4D:** és un programari de disseny 3D desenvolupat per Maxon. És utilitzat tant per professionals com per principiants a causa de la seva interfície intuïtiva i fàcil d'emprar. Cinema 4D s'utilitza en la indústria del cinema, la televisió i els mitjans digitals per crear efectes visuals, animacions i gràfics en moviment. Dos exemples del seu potencial són les pel·lícules Pacific Rim i Tron: Legacy.
- **Houdini:** és un programari que destaca pel seu enfocament processal, cobrint totes les àrees de la producció 3D. És responsable de plans de pel·lícules tan populars com Frozen i Zootopia.

Aquests són només alguns exemples dels programes de disseny 3D privatius més utilitzats. Cadascun té les seves pròpies característiques i s'utilitza segons necessitats específiques.

Impressió 3D: La materialització dels objectes virtuals

La impressió 3D ha revolucionat encara més el disseny 3D en permetre la materialització física dels objectes virtuals creats. És a dir, podem emprar programes de disseny 3D per crear models digitals que es poden imprimir en 3D. En utilitzar tecnologies de fabricació additiva, les impressores 3D poden transformar els dissenys en objectes tangibles de manera ràpida i precisa. Això ha ampliat encara més les possibilitats d'aplicació del disseny 3D en diversos camps, com la creació de prototips, la medicina regenerativa, l'arquitectura sostenible o la personalització de productes.

Conclusió

El disseny 3D s'ha convertit en una eina essencial a causa de la seva capacitat per visualitzar idees de manera eficient i precisa. Tant les eines lliures com les privatives exerceixen un paper important en aquest procés, ja que brinden opcions adaptades a les seves necessitats i recursos disponibles. Amb la integració de la impressió 3D, el disseny 3D ha fet un pas més enllà en permetre la materialització dels objectes virtuals i ha obert noves possibilitats i oportunitats en multitud d'àmbits.

Saber-ne més

Un exemple de la important relació que existeix entre el disseny 3D i la impressió 3D dels materials dissenyats la podem trobar en els treballs d'Ayúdame3D. Ayúdame3D és una entitat espanyola que fomenta el valor social de la tecnologia a través de programes de conscienciació tecnològic-social amb la finalitat d'ajudar a col·lectius vulnerables de tot el món. Gràcies a això crea i lliura braços impresos en 3D, denominats tresdesis, de manera gratuïta a persones amb discapacitat.

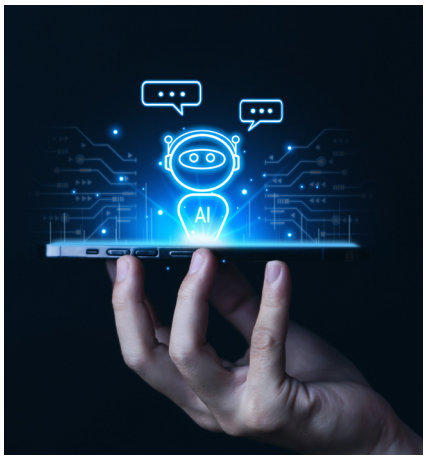
ayudame3d.org

Creació de continguts digitals

Nivell C1 3.1 Desenvolupament de continguts

Ús de la intel·ligència artificial en el núvol per crear nous vídeos

Ús de la intel·ligència artificial en el núvol per crear nous vídeos



La creació de vídeos amb intel·ligència artificial (IA) és un camp en constant evolució, emergint noves tècniques i eines tot el temps. L'ajuda de la IA s'està convertint en un component integral de la producció de mitjans digitals, ja que aquesta manera de crear i editar vídeos ofereix oportunitats per estalviar temps, augmentar l'eficiència i obrir noves possibilitats creatives.

Usos habituals de IA a l'hora de generar vídeos

Entre els usos actuals que posseeixen aquestes utilitats es troben:

1 | Automatització del procés d'edició

Les eines d'edició de vídeo impulsades per IA permeten als usuaris editar vídeos de la mateixa manera que editarien un document de text. En generar transcripcions automàtiques del contingut del vídeo, els usuaris poden eliminar fàcilment les parts no desitjades, simplement eliminant el text corresponent de la transcripció.

2 | Millora de la qualitat del vídeo

Gràcies a aquests instruments es poden fer tasques com eliminar àudio, el seguiment del moviment, l'eliminació del fons i la detecció de silencis. Aquestes funcions permeten als usuaris millorar la qualitat del seu contingut sense necessitat de tenir habilitats tècniques avançades.

3 | Creació de contingut generatiu

S'hi pot crear nou contingut a partir de text o imatges, basant-se en les indicacions donades per l'usuari, obrint noves possibilitats per a la creació de continguts visuals únics.

4 | Personalització i brànding

Les eines de vídeo de IA també poden ajudar les empreses a personalitzar els seus vídeos per reflectir la seva marca o crear vídeos utilitzant avatars digitals. Això pot ser particularment útil per a la creació ràpida de vídeos explicatius, vídeos de capacitació o anuncis, sense la necessitat de contractar actors o personal de càmera.

Eines de generació de vídeo

A continuació, presentem alguns exemples d'eines disponibles que pots emprar de manera gratuïta o de pagament.

1 | Descript

Et permet generar una transcripció de tot el que dius juntament amb un conjunt d'escenes, separant la pista de vídeo automàticament. Pots ressaltar parts de la transcripció que vols eliminar, i Descript les edita per tu. A més, pots dividir el teu vídeo en escenes i agregar material d'arxiu d'alta qualitat al teu projecte sense sortir de la finestra de l'editor.

DESCRIPT
descript.com

2 | Wondershare Filmora

Aquesta és una eina d'edició de vídeo tradicional amb utilitats basades en IA. Algunes de les funcionalitats basades en IA inclouen retallada intel·ligent, eliminació del soroll d'àudio, estirament d'àudio, seguiment de moviment, eliminació de fons i detecció de silenci. També pots fer canvis en la configuració del teu vídeo en exportar, triant el format, la resolució, la qualitat i fins i tot la velocitat de fotogrames. A més, té agregat ChatGPT, per la qual cosa també es pot generar guions, subtítols, descripcions, entre d'altres, mitjançant IA.

WONDERSHARE FILMORA
filmora.wondershare.net/es

3 | Runway

Runway ofereix característiques de IA per arrodonir vídeos i generar de text a vídeo, d'imatge a imatge i la possibilitat d'entrenar els teus models de IA per a la generació d'imatges. També ofereix característiques com a canvi de fons de pantalla verda (croma), eliminació i reemplaçament d'objectes i reemplaçament de seccions d'imatges.

RUNAWAY
runwayml.com

4 | Peech

Està dissenyada per a equips de màrqueting de contingut. Et permet afegir el teu kit de marca perquè Peech pugui assenyalar automàticament tots els teus vídeos. Cada vegada que afegeixes un nou vídeo, Peech afegeix aquests elements juntament amb subtítols personalitzables.

PEECH

peech-ai.com

5 | Synthesia

Aquesta eina et permet generar vídeos utilitzant avatars digitals de IA. Suporta múltiples idiomes, però els avatars poden no ser completament creïbles quan es veuen en pantalles grans.

SYNTHESIA

synthesia.io

CARACTERÍSTIQUES DE GENERADORS DE VÍDEO MITJANÇANT IA

Nom	Indicat per	Plataforma	Gratuïtat
Descript	Edició de vídeo mitjançant l'edició del guió	Windows, Mac (Web per a algunes funcions)	Sí, 1 hora de transcripció i marca de gua
Wondershare Filmora	Polir vídeo amb eines de IA	Windows, Mac, iOS, Android	Sí, marca d'aigua
Runway	Experimentar amb IA generativa	Web	Sí, amb 125 crèdits, 3 projectes
Peech	Equips de màrqueting de continguts	Web	Sí, 2 vídeos/mes
Synthesia	Ús d'avatars digitals	Web	No

Saber-ne més

Existeixen cursos en línia (Coursera, machine learning A-Z, etc.) que proporcionen coneixements detallats sobre IA, les seves eines, tecnologia i tendències. No obstant això, depenent de l'eina específica que vulguis aprendre, és possible que necessitis cercar tutorials o cursos específics per a aquesta eina. En el següent enllaç trobaràs les millors alternatives existents en el moment de la redacció d'aquest document.

unite.ai/best-ai-vídeo-generators

Creació de
continguts digitals

Nivell C1 3.1 Desenvolupament de continguts

El teu lloc web i els seus continguts digitals són accessibles?

El teu lloc web i els seus continguts digitals són accessibles?

Protocol per avaluar si el teu lloc web i els teus continguts són accessibles

Una vegada familiaritzats amb els criteris exposats en els mòduls anteriors, es proposa un protocol o procediment a seguir per comprovar si un lloc web compleix amb els principals requisits d'accessibilitat (que afectin text, imatge, vídeo i so).

1 | Verificació de text:

- Verifica que la grandària de font sigui llegible i es pugui ajustar segons les preferències de l'usuari.
- Comprova que no s'utilitzin massa colors o colors pastel o molt cridaners per transmetre la informació, ja que això pot dificultar la comprensió per a persones amb discapacitats visuals o daltonisme.
- Examina l'estructura d'encapçalats adequada (H1, H2, etc.) per facilitar la navegació i la comprensió del contingut.
- Assegura't que existeix un alt contrast entre el color del text i el color del fons.
- Tria una tipografia senzilla que es llegeixi amb facilitat, perquè l'elecció d'una amb excessius adorns pot dificultar la lectura.

2 | Verificació d'imatges

- Assegura't que totes les imatges importants tenguin text alternatiu (atribut ALT) descriptiu. Això permetrà que les persones amb discapacitat visual comprenguin el contingut de les imatges a través de lectors de pantalla o altres tecnologies d'assistència.
- Comprova que els enllaços o botons que utilitzin només imatges tenguin text alternatiu o etiquetes de títol explicatives.

3 | Verificació de vídeos:

- Assegura't que els vídeos incloguin subtítols per a persones amb discapacitat auditiva. Els subtítols han de ser precisos, sincronitzats i representar adequadament el diàleg i els efectes de so rellevants.

- Verifica que es proporcioni una versió de vídeo amb àudio descriptiu per a persones amb discapacitat visual. L'àudio descriptiu és una narració addicional que descriu les accions visuals importants que ocorren en el vídeo.
- Comprova que els controls de reproducció de vídeo siguin accessibles i es puguin operar amb facilitat utilitzant el teclat.
- Valora si és possible incorporar un vídeo superposat d'una persona o avatar emprant la llengua de signes.

4 | Verificació de so:

- Assegura't que els elements d'àudio tenguin controls de reproducció que permetin aturar, detenir o ajustar el volum del so.
- Comprova que es proporcioni una alternativa de text o una transcripció per al contingut d'àudio, cosa que permet que les persones amb discapacitat auditiva accedeixin al contingut.

5 | Proves de compatibilitat amb tecnologia d'assistència

- Realitza proves utilitzant tecnologia d'assistència, com a lectors de pantalla (com JAWS, NVDA o VoiceOver) i teclats en lloc de ratolins. Això et permetrà avaluar com es presenta i s'interactua amb el contingut del lloc web per a persones amb discapacitats visuals o de mobilitat.

6 | Verificació de navegació i estructura:

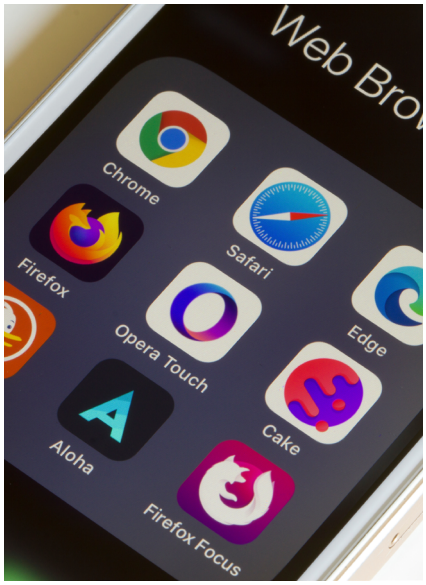
- Assegura't que la navegació del lloc web sigui clara, coherent i fàcil d'entendre. Utilitza etiquetes adequades per als elements de navegació i proporciona indicadors visuals clars de la ubicació actual de l'usuari en el lloc.
- Comprova que s'utilitzin enllaços descriptius que indiquin clarament el destí de l'enllaç, en lloc d'enllaços genèrics com a "clic aquí".
- Verifica que l'ordre de tabulació sigui lògic i que es pugui navegar pel lloc web utilitzant només el teclat, sense dependre de l'ús del ratolí.
- Assegura't que els formularis siguin accessibles, i que hi incloguin etiquetes clares i associacions adequades entre els camps d'entrada i les etiquetes corresponents.

7 | Verificació de contrast de color:

- Assegura't que el contrast de color entre el text i el fons sigui suficient per facilitar la llegibilitat. Verifica que es compleixin els criteris de contrast establerts en les pautes d'accessibilitat web.
- Comprova que els elements interactius, com els botons o enllaços, tenguin un contrast adequat per facilitar la seva identificació i ús.

8 | Proves de compatibilitat amb diferents navegadors i dispositius:

- Fes proves d'accessibilitat en diferents navegadors web populars, com Chrome, Firefox, Safari i Edge, per assegurar-te que el lloc web sigui accessible en totes les plataformes.
- Verifica l'accessibilitat en diferents dispositius, com computadores d'escriptori, tauletes i dispositius mòbils, per garantir una experiència accessible en totes les pantalles.



9 | Validació del codi:

- Utilitza eines de validació d'HTML i CSS per assegurar-te que el codi del lloc web compleixi amb els estàndards i les millors pràctiques recomanades. Un codi net i ben estructurat és fonamental per a l'accessibilitat.

10 | Proves de rendiment:

- Du a terme proves de rendiment del lloc web per a garantir una càrrega ràpida i una experiència fluida. Els retards en la càrrega poden afectar negativament les persones amb discapacitats, especialment a aquells que utilitzen tecnologies d'assistència.

11 | Avaluació d'errors de llenguatge i llegibilitat:

- Verifica que no hi hagi errors de gramàtica, ortografia o puntuació que puguin dificultar la comprensió del contingut per part de les persones amb discapacitats cognitives o de lectura.
- Assegura't que l'estructura de les oracions i els paràgrafs sigui clara i concisa per facilitar la lectura i comprensió.

12 | Proves d'interactivitat i funcionalitat:

- Verifica que els elements interactius, com a formularis, botons i menús desplegables, siguin accessibles i es puguin operar fàcilment utilitzant només el teclat.
- Assegura't que les animacions i les actualitzacions dinàmiques de contingut siguin accessibles i no provoquin distracció o confusió per als usuaris.

13 | Verificació de l'etiqueta d'idioma:

- Assegura't que el lloc web utilitzi l'etiqueta d'idioma HTML correctament. Això és especialment important per a les persones que utilitzen tecnologies d'assistència, ja que els permet adaptar la pronunciació i el llenguatge del contingut.

14 | Avaluació de la navegació per tabulació:

- Comproveu que l'ordre de tabulació dels elements interactius sigui lògic i predictible. Això garanteix que les persones que utilitzen el teclat per navegar puguin accedir i operar correctament tots els elements de la pàgina.

15 | Proves de contrast en diferents nivells de zoom:

- Comprova que l'ordre de tabulació dels elements interactius sigui lògic i predictible. Això garanteix que les persones que utilitzen el teclat per a navegar puguin accedir i operar correctament tots els elements de la pàgina.

16 | Verificació de la llegibilitat del contingut en diferents grandàries de pantalla:

- Avalua la llegibilitat del contingut en diferents grandàries de pantalla, assegurant-te que el text sigui prou gran i llegible en dispositius mòbils i pantalles més petites.

17 | Proves de teclat:

- Verifica que tots els elements i accions interactives en el lloc web puguin ser accedits i utilitzats a través del teclat. Això és essencial per a les persones que no poden utilitzar un ratolí o altres dispositius d'entrada similars.

18 | Proves de redimensionament de text:

- Assegureu-vos que el contingut del lloc web es pugui redimensionar fins al 200% sense que es perdi informació, funcionalitat o disseny. Això és especialment important per a les persones amb discapacitats visuals que necessiten augmentar la mida del text per llegir-lo amb claredat.

19 | Validació de formularis i missatges d'error:

- Verifica que els formularis en el lloc web incloguin etiquetes clares, missatges d'error descriptius i validació adequada per facilitar la interacció i la correcció d'errors per part dels usuaris.

20 | Documentació de troballes i millores:

- Documenta totes les troballes, errors i millores identificats durant el procés de verificació d'accessibilitat. Registra els canvis implementats i fa un seguiment de les actualitzacions realitzades en el lloc web per millorar la seva accessibilitat.

L'accessibilitat web és un procés continu i dinàmic. Mantingues un pla d'acció per abordar les barreres d'accessibilitat identificades i fa actualitzacions regulars per millorar l'accessibilitat del lloc web sobre la base de les noves pautes i estàndards que es publiquin.

Llocs web que avaluen automàticament altres webs

Per avaluar si un lloc web compleix amb els criteris d'accessibilitat necessaris, existeixen webs que fan anàlisis automàtiques d'altres webs a través del seu URL. Alguns exemples són:

- **WebAIM's WAVE** (*Web Accessibility Evaluation Tool*)**:** WAVE és una eina en línia gratuïta que proporciona anàlisi automatitzada d'accessibilitat web. Simplement, ingressa l'URL d'un lloc web i WAVE generarà un informe detallat que destaca els problemes d'accessibilitat detectats.

WEBAIM'S WAVE

wave.webaim.org

- **Axe by Deque Systems:**
  Axe és una *suite* d'eines d'accessibilitat desenvolupada per Deque Systems. Proporciona una extensió per a navegadors web que permet analitzar llocs web a la recerca de problemes d'accessibilitat. També s'ofereix una API per a la integració en fluxos de treball de desenvolupament.

AXE BY DEQUE SYSTEMS
deque.com/axe

- **Lighthouse by Google:**
  Lighthouse és una eina de codi obert desenvolupada per Google que ofereix auditories automàtiques per a la qualitat de les pàgines web. A més de les auditories generals, Lighthouse inclou un conjunt d'auditories específiques per a l'accessibilitat web.

LIGHTHOUSE BY GOOGLE
e.digitall.org.es/lighthouse

- **Tenon.io:**
  Tenon.io és una plataforma d'anàlisi d'accessibilitat web que permet analitzar llocs web a la recerca de problemes d'accessibilitat. Ofereix una API per a la integració amb fluxos de treball de desenvolupament i també proporciona una interfície d'usuari en línia per a fer anàlisis ràpides.

TENON.IO
tenon.io

Saber-ne més

Si vols saber més sobre els estàndards internacionals sobre accessibilitat, pots consultar el W3C Web Accessibility Initiative (WAI).

e.digitall.org.es/wai

Creació de
continguts digitals

Nivell C1 3.1 Desenvolupament de continguts

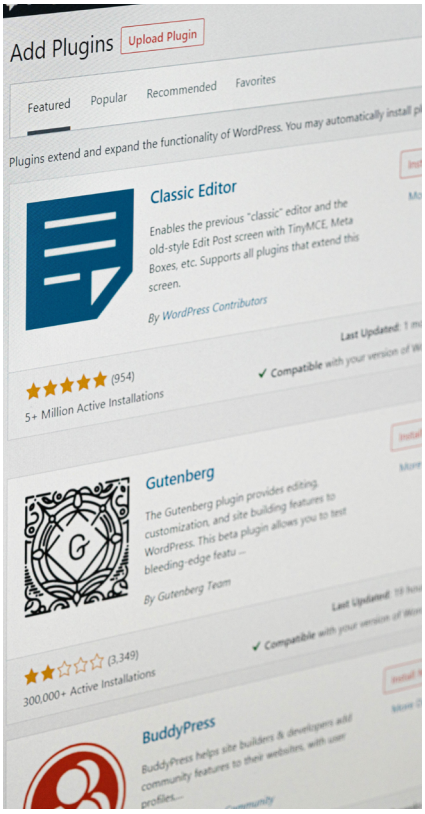
Augmentant la funcionalitat dels gestors de continguts per a la creació de llocs web

Augmentant la funcionalitat dels gestors de continguts per a la creació de llocs web

Introducció

Els gestors de continguts exerceixen un paper fonamental en la creació de llocs web, cosa que permet als usuaris administrar i publicar el seu contingut de manera eficient. No obstant això, la funcionalitat bàsica d'aquests gestors pot no ser suficient per satisfer totes les necessitats d'un lloc web en particular. És aquí on entren en joc els **complements** o ***connectors***, eines que poden instal·lar-se en els gestors de continguts per a augmentar la seva complexitat i capacitat. O, dit d'una altra forma, a través dels connectors podrem adaptar el nostre lloc web a les nostres necessitats particulars. En aquest text, explorarem la importància d'aquests complements i examinarem exemples d'ús amb un dels gestors de continguts més populars, WordPress.

Augmentant la funcionalitat amb *connectors*

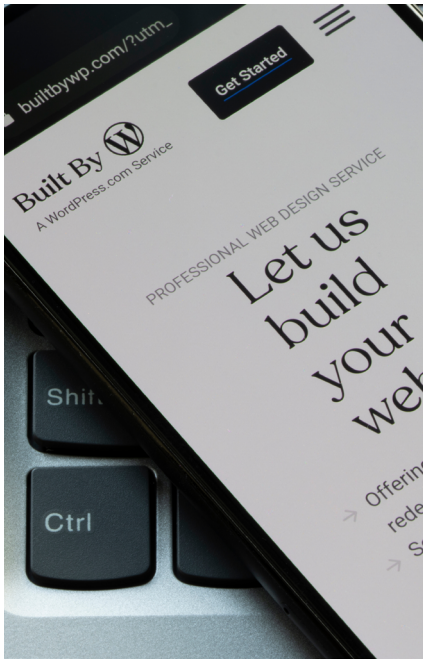


Els complements o connectors són petites aplicacions que s'integren amb els gestors de continguts, agregant noves funcionalitats i característiques. Aquestes eines ofereixen una manera convenient de personalitzar i ampliar les capacitats d'un gestor de continguts segons les necessitats de l'usuari. Els **complements** poden variar des de la simple incorporació de formularis de contacte fins a la creació de complexos sistemes de comerç electrònic.

Descriurem a continuació dos tipus generals de *connectors*, els de **pagament** i els **gratuïts**. En general, els connectors de pagament ofereixen funcionalitats més avançades i especialitzades. Aquestes solucions solen ser desenvolupades per experts i empreses que inverteixen temps i recursos en el seu desenvolupament. Els complements de pagament solen oferir un suport tècnic més sòlid, actualitzacions regulars i característiques exclusives.

D'altra banda, també hi ha una àmplia gamma de complements gratuïts disponibles per als gestors de continguts. Aquests complements són desenvolupats per una comunitat de programadors i dissenyadors que contribueixen de manera voluntària. Encara que els complements gratuïts poden no oferir tantes funcions com els de pagament, poden ser una excel·lent opció per als qui tenen un pressupost limitat o necessiten funcionalitats bàsiques.

Ús de connectors a WordPress



En l'univers de WordPress, existeixen *connectors* per a pràcticament qualsevol funcionalitat que puguem imaginar. Igual que comentem en nivells anteriors quan descrivim les característiques de WordPress, els connectors ens permetran aquesta personalització sense que necessitem coneixements de programació.

La instal·lació de *connectors* a WordPress és un procés senzill que es pot fer mitjançant dos mètodes: bé a través del repositori de WordPress o bé a través d'una instal·lació manual. En aquest text descriurem com instal·lar connectors a través del repositori de WordPress.

Per instal·lar un *connector* en WordPress seleccionarem des del nostre panell d'administració la pestanya "*connectors*" en el menú lateral (Figura 1). Podem navegar per les diferents categories de connectors, realitzar cerques per paraules clau o explorar les llistes dels més populars o recomanats. Cada connector en el repositori té la seva pròpia pàgina de detalls, on es proporciona informació sobre la seva funcionalitat, característiques, requisits i ressenyes d'usuaris.

Una vegada hem trobat el connector a instal·lar, només hem de prémer a "*Instal·lar ara*" i automàticament WordPress descarregarà i instal·larà el connector en el nostre lloc web. En la Figura 2 veiem com a exemple el connector "*Assessor de Cookies RGPD per a normativa europea*" que ens permetrà adaptar-nos al Reglament General de Protecció de Dades.

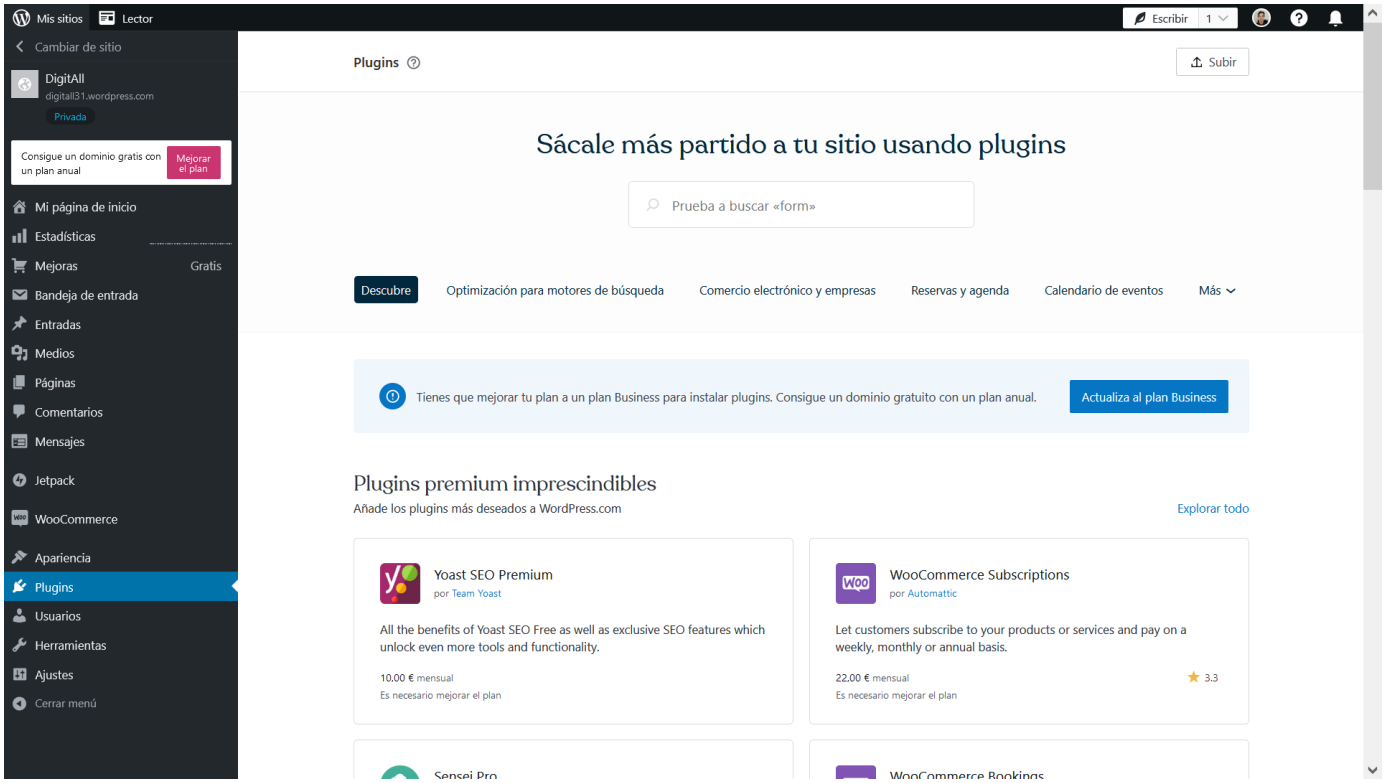


Figura 1. Visió general del menú "Plugins" en el panell d'administració de WordPress.

Figura 2. Detall del connector "Assessor de Cookies RGPD per a normativa europea". Podem instal·lar-ho en la nostra web simplement prement el botó de baix a la dreta i ens ajudarà a generar el famós botó d'"Acceptar cookies".

Com veiem en la Figura 1, una de les opcions és cercar per categories populars. Algunes són:

- **Optimització per a motors de cerca (SEO):** per optimitzar la visibilitat i el posicionament dels llocs web en els motors de cerca.
- **Comerç electrònic i empreses:** per crear botigues en línia i gestionar pagaments.
- **Seguretat:** per protegir el lloc web contra atacs i amenaces.
- **Formularis:** per crear formularis de contacte personalitzats i enquestes.
- **Galeries d'imatges:** per mostrar imatges de manera atractiva en el lloc web**.**

Connectors populars a WordPress

Alguns dels connectors gratuïts més populars per a Wordpress són *Yoast SEO, Contact Form 7, Elementor Website Builder, WooCommerce o Akismet Anti-Spam*. D'altra banda, alguns dels connectors de pagament més populars per a Wordpress són les versions de pagament de Yoast i els connectors per gestionar un comerç electrònic basat en *WooCommerce*.

Conclusió

Els gestors de continguts són eines poderoses per a la creació de llocs web, però a vegades la seva funcionalitat bàsica no és suficient per satisfer totes les necessitats. És aquí on els connectors entren en joc, cosa que permet als usuaris ampliar les capacitats dels gestors de continguts de manera fàcil i personalitzada. Sigui optant per complements de pagament o gratuïts, la varietat d'opcions disponibles en l'univers de WordPress permet als usuaris personalitzar els seus llocs web i millorar l'experiència tant per a ells com per als visitants. Explorar i experimentar amb connectors és una manera efectiva d'augmentar la funcionalitat dels gestors de continguts i portar els llocs web al següent nivell.

Saber-ne més

Pots aprendre molt sobre com funcionen connectors específics en el directori de connectors de WordPress en espanyol. Aquest és un lloc web que conté una gran quantitat de connectors gratuïts i alguns de codi obert per a WordPress. Per a cada connector pots veure tant les seves característiques, les valoracions d'altres usuaris, la compatibilitat amb la versió actual de WordPress instal·lada, així com el nombre d'instal·lacions actives. Atès que el repositori de WordPress és mantingut i administrat per l'equip de Wordpress.org, tots els connectors inclosos en el repositori són revisats i aprovats per l'equip de WordPress abans de ser publicats, la qual cosa ajuda a garantir la qualitat i seguretat.

es.wordpress.org/plugins

DigitAll

Creació de
continguts digitals

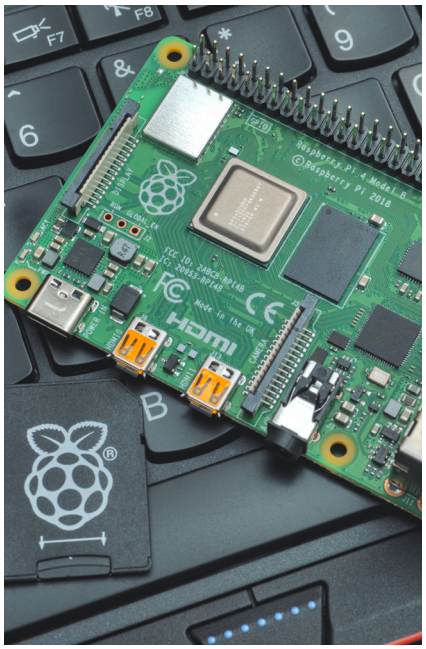
3.2
INTEGRACIÓ I REELABORACIÓ DE CONTINGUT DIGITAL

Creació de continguts digitals

Nivell C1 3.2 Integració i reelaboració de contingut digital

Tipus de sensors i actuadors

Tipus de sensors i actuadors



Sensors

La construcció d'un sistema programari-maquinari requereix l'ús de microcontroladors entre els quals destaquen Arduino i l'ordinador Raspberry Pi. L'elecció d'una mena de microcontrolador depèn del projecte electrònic a construir. L'avantatge d'Arduino és que executa immediatament la tasca. Raspberry Pi, per disposar de més potència de còmput i un sistema operatiu, és més adequat per ser emprat com un microordinador funcional portàtil. Tots dos tipus de plaques permeten la integració en el sistema d'altres dispositius anomenats sensors i actuadors. La figura núm. 1 mostra un exemple d'aquests dispositius electrònics.

Els sensors són dispositius que són capaços de mesurar magnituds físiques o químiques, denominades **variables d'instrumentació**, i transformar-les en **variables elèctriques** mitjançant un **transductor**.

La conversió de la variable d'instrumentació en variable elèctrica es pot fer mitjançant una resistència tèrmica (per exemple, la temperatura), un fotodíode (intensitat de llum) o mitjançant un cristall piezoelèctric (pressió) que adquireix polaritat quan és sotmès a una pressió mecànica. Les variables elèctriques poden ser resistència, capacitat, voltatge, corrent, etc. Les sortides del sensor poden ser analògiques, digitals o connectades a través de busos de comunicació.

Els sensors fan mesuraments i aquest procés està caracteritzat per diversos paràmetres que es descriuen a continuació. El **domini** és el conjunt de possibles valors que el dispositiu és capaç de mesurar. La **resolució** és la mínima variació de la variable d'instrumentació que el sensor pot detectar. La **precisió** és el màxim error que pot cometre. Les derives són altres magnituds, a més de la desitjada, que afecten les mesures resultants. La **desviació** de zero (offset) és el valor de la variable elèctrica quan la variable d'instrumentació és nul·la. A continuació, s'enumeren els principals tipus de sensors:

Tipus	Descripció	Característiques
Desplaçament i distància	Mesuren desplaçaments lineals, posicions o distàncies lineals. El rang d'aquests dispositius va des de mil·límetres a centenars de metres.	Existeixen amb contacte físic (amb fregament) i sense contacte que es basen en làser o ultrasons.
Acceleració	Mesuren l'acceleració o la vibració des d'uns pocs g's fins a milers de g's.	Existeixen de diversos tipus com piezoresistius, piezoelèctrics i acceleròmetres capacitatius.
Giroscopis i referències inercials	Permeten mesurar o mantenir l'orientació en l'espai quan es produeix el moviment d'un objecte. Les referències inercials són sistemes de mesura de posició i velocitat angular. Aquest tipus de sensors es denominen sensors giroscòpics.	Integren giroscopis i acceleròmetres triaxials.
Sensors de parell i torsió	Mesuren la força de torsió a la qual se sotmet un eix. S'empren en l'estudi d'elements de rotació.	Són estàtics o dinàmics. El transductor transforma la torsió en una variació de voltatge.
Pressió i cabal	Transformen una força per unitat de superfície en un voltatge equivalent. Destaquen per a mesurar la pressió de líquids com a aigua, oli, o líquids de frens. Els de cabal mesuren la quantitat de material (en pes o volum) que circula per un canal per unitat de temps.	Tenen aplicacions en la construcció d'interruptors de pressió o monitoratge de nivells.
Paràmetres ambientals	Permeten mesurar la temperatura, humitat, pressió baromètrica, CO, NOx, PM10, PM2.5, etc.	Tenen aplicacions en agricultura per al control intel·ligent del reg o per al control d'emissions en ciutats.
Sensors de presència	Detecten la presència d'un objecte i la seva proximitat amb el punt de referència. Tenen aplicacions en vehicles no tripulats o vigilància autònoma.	Consten d'un parell emissor/receptor. S'usen en aplicacions per automàticament obrir/tancar portes, encendre/apagar llums, encesa automàtica de pantalles.
Sensors acústics	Té aplicacions en la detecció de comandaments de veu o en la vigilància intel·ligent.	Transformen ones acústiques en impulsos elèctrics.
Sensor de llum	Tenen una superfície sensible a la llum, produint valors diferents de resistència.	Es poden utilitzar per mesurar el nivell de llum i poder detectar si és de dia o de nit o corregir-ne la il·luminació.

La Figura 1 mostra un exemple d'un dispositiu Raspberry Pi connectat a un petit circuit elèctric que integra dos sensors mediambientals que mesuren gasos GPL i monòxid de carboni, respectivament. La Figura 2 mostra el resultat real.

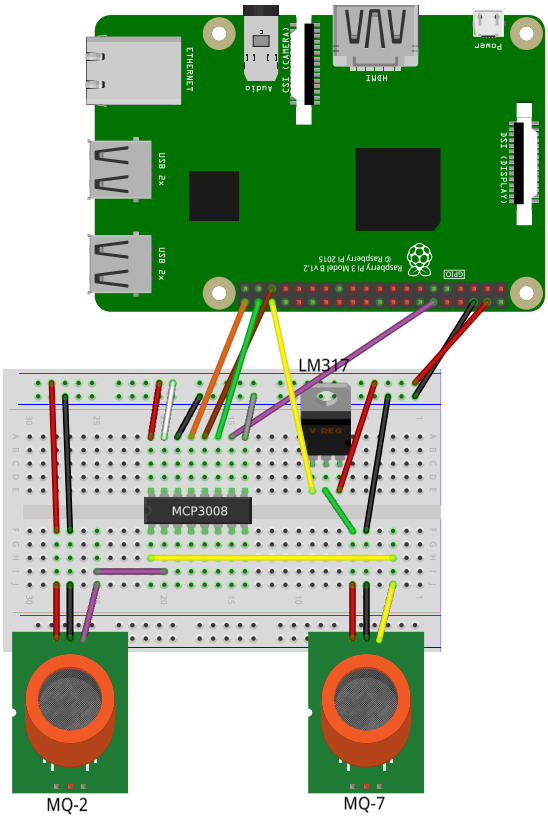


Figura 1. Exemple de dispositiu Raspberry Pi connectat a dos sensors de paràmetres ambientals.

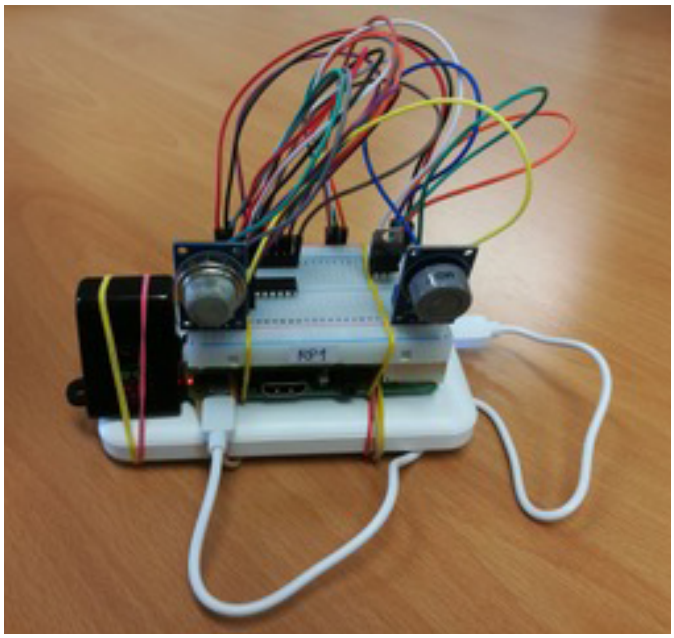


Figura 2. Resultat real de l'esquema mostrat a la Figura 1.

Actuadors

Els actuadors són dispositius que transformen energia hidràulica, pneumàtica o elèctrica en l'activació d'un procés que genera una acció sobre un element extern al sistema.

Per exemple, un actuador pot rebre una ordre d'un microcontrolador, com Arduino o Raspberry Pi, i posar en funcionament el motor d'una bomba.

La següent taula resumeix els principals tipus d'actuadors:

Tipus	Descripció	Exemples d'ús
Mecànics	Transformen l'acció rotativa d'una entrada en un moviment lineal.	Gats mecànics. S'empren per a la col·locació exacta i repetible d'objectes.
Pneumàtics	Transformen aire comprimit en una força motriu que produeix una acció mecànica.	Pinces pneumàtiques que permeten recollir objectes d'un lloc i deixar-los en un altre lloc.
Hidràulics	Empren una força hidràulica per a empènyer i obtenir una força externa. S'usen quan es necessita una potència elevada.	S'empren en grues i en excavadores, així com per a l'automatització de vàlvules.
Elèctrics	La font de la força és l'energia elèctrica.	Vàlvules multivoltes, motors, o relés.
Tèrmics	La font de la força és l'energia elèctrica.	Automatització de sistemes de calefacció i refrigeració.

Els sensors van connectats a les entrades de les plaques d'Arduino o de la Raspberry Pi mentre que els actuadors es connecten en les sortides. A més, es poden connectar els perifèrics que són dispositiu maquinari que permeten emmagatzemar o intercanviar informació amb l'exterior del sistema. Exemples de perifèrics són les pantalles LCD, teclats, memòries externes, impressores, càmeres, micròfons, visualitzadors numèrics, etc.

El document "***Paradigmes de la programació. Visió general***" (competència 3.4 del Nivell C1) és clau per al control de sensors i actuadors en diferents camps, com la robòtica i l'automatització industrial. Mitjançant l'ús de llenguatges de programació, es poden crear programes que interactuin amb els sensors per recopilar informació de l'entorn i enviar-la als actuadors per dur a terme accions específiques. A més, la informació capturada pels sensors es pot utilitzar per generar contingut multimèdia, com a imatges i vídeos, que proporcionin una representació visual de les dades recopilades. Això és útil en aplicacions de monitoratge ambiental o de seguretat.

PARADIGMES DE LA PROGRAMACIÓ. VISIÓ GENERAL

Document referenciat: ***A3C34C1D01***

⚠ ATENCIÓ

A l'hora de triar un sensor, hem de revisar detingudament les característiques i triar un que sigui compatible amb la nostra placa (tensió/voltatge i intensitat de corrent/amperatge) i que el seu domini sigui conforme a les magnituds del nostre sistema. Un altre paràmetre que influeix en l'elecció és la seva facilitat d'ús, disponibilitat de llibreries i documentació.

Creació de continguts digitals

Nivell C1 3.2 Integració i i reelaboració de contingut digital

Autenticitat i verificació de continguts digitals

Autenticitat i verificació de continguts digitals

En l'actualitat, el món viu en la denominada societat de la informació. Les persones poden crear, modificar, accedir i compartir la informació i el coneixement per millorar els seus processos productius i, en definitiva, la seva qualitat de vida.

La **societat de la informació** és aquella en la qual les tecnologies de la informació i la comunicació (TIC) permeten **l'accés**, la **creació** i la **distribució** de la informació en les activitats econòmiques, socials i culturals.

Paradoxalment, una de les conseqüències de la societat de la informació és la desinformació. L'accés a la informació no implica necessàriament estar informat, especialment quan la informació és ingent, heterogènia (qualsevol usuari pot produir contingut) i, sovint, contradictòria. En aquest sentit, la societat de la informació planteja als ciutadans el repte de filtrar i discriminar amb sentit crític la informació útil i veraç d'aquella que no ho és.

L'**anàlisi forense informàtic** és l'àrea de les ciències de la computació que engloba el conjunt de tècniques que permeten **identificar, preservar, recuperar i analitzar** mitjans digitals (maquinari i programari).

Existeixen multitud d'eines d'anàlisi forense informàtic encaminades a la presa de decisions, així com a la presentació de fets i evidències respecte a la informació digital. Per exemple, ***FotoForensics*** (***fotoforensics.com***), que permet fer diferents anàlisis sobre una imatge provinent d'un URL o del dispositiu local per comprovar si ha estat o no modificada, ***AmpedAuthenticate*** (***ampedsoftware.com***) en la qual, prenent una imatge com a referència, és possible dur a terme diferents anàlisis i revelar l'historial de processament d'una imatge digital per determinar si es tracta o no de contingut manipulat, o ***ForensicToolkit (FTK)*** (***exterro.com/forensic-toolkit***), que ofereix una solució integrada que permet recuperar la

informació d'un dispositiu digital i dur a terme l'anàlisi forense del seu contingut.

Un exemple clar de contingut que provoca desinformació són les **notícies falses**. Es tracten de notícies falses publicades en la xarxa amb l'objectiu d'enganyar, manipular i induir a error als lectors a més d'enaltir o desprestigiar a una persona o institució. Aquestes notícies es presenten amb aparença de veracitat en mitjans digitals, especialment en xarxes socials.

Recentment, apareixen en xarxes socials i nombrosos mitjans digitals continguts audiovisuals falsos. Alguns d'aquests continguts tenen un propòsit humorístic, però altres tenen objectius similars als de les notícies falses. En aquests continguts, coneguts com hipertrucatges, destaquen vídeos en els quals s'integren cares i veus de personatges reals per elaborar continguts falsos.

Hipertrucatge, també conegut com a falsificació profunda, engloba a aquells continguts digitals com a imatges i vídeos ficticis generats a través de **tècniques d'intel·ligència artificial.**

Per a la generació d'hipertrucatges s'utilitza una tècnica d'intel·ligència artificial, concretament d'aprenentatge profund, basada en xarxes neuronals i anomenada **Xarxa Generativa Antagònica** (GAN). Hi coexisteixen dues xarxes. Una xarxa s'encarrega d'aprendre d'una gran base de dades d'imatges i vídeos existents per generar altres nous. La segona xarxa s'encarrega de discriminar si les imatges i vídeos generats per la primera són falsos o no. Quan un contingut generat per la primera xarxa no és detectat com a fals per la segona, l'hipertrucatge està preparat.

Per tot l'anterior, garantir l'autenticitat d'un contingut digital i verificar si ha estat manipulat o no és fonamental per a evitar la desinformació i altres delictes relacionats amb el frau, l'honor o la imatge. A vegades, és possible **detectar que el contingut ha estat manipulat a través de l'observació d'incorreccions en imatges i vídeos**. Per a una detecció més formal i rigorosa, a més de les eines d'anàlisi forense vistes amb anterioritat, existeix tecnologia antihipertrucatge que tracta de detectar-

los per autenticar el contingut i també evitar que un contingut s'utilitzi per generar nous hipertrucatges.

No obstant això, existeixen molts més recursos destinats a la tecnologia per crear **hipertrucatges** que a les eines per detectar-los. A més, els desenvolupadors d'aquests continguts aprofiten les recerques publicades sobre com detectar-los per millorar la seva tecnologia i continuar generant contingut fals que fuita a aquests sistemes. Per aquest motiu, el programari disponible per a la detecció d'hipertrucatges no està generalment obert als usuaris.

Entre el programari **antihipertrucatge** més utilitzat i disponible per al públic general, es troben els següents:

- **Sensity:** es tracta d'una plataforma web dedicada a la detecció de contingut fals. Permet pujar arxius en diferents formats, inclosos documents, per detectar si es tracta de **contingut fraudulent**. Entre les seves eines disponibles, destaquen la verificació d'identitat, detecció d'hipertrucatge, reconeixement facial, o reconeixement de documents fraudulents. Per accedir a les eines, s'ha de completar un formulari de contacte amb la companyia per sol·licitar l'accés.
- **Deepware Scanner**: es tracta d'un projecte de codi obert per a la detecció d'hipertrucatges. La seva execució es basa en una interfície de línia de comandaments i permet escanejar els vídeos d'un directori del dispositiu per detectar quins són hipertrucatge. Al contrari que Sensity, el sistema del qual es basa en **RGA**, **Deepware Scanner** utilitza models basats en xarxes neuronals convolucionals. En tractar-se d'un projecte de codi obert, és possible conèixer com està dissenyat aquest sistema, així com consultar el codi font dels models preentrenats que s'utilitzen per a la detecció d'hipertrucatges.
- **Fake Profile Detector:** és una extensió gratuïta de Google Chrome que permet identificar si una imatge es tracta o no d'hipertrucatges. La seva instal·lació és molt senzilla i per utilitzar-la simplement cal fer clic amb el botó dret del ratolí

NOTA

Algunes de les comprovacions visuals que es poden realitzar per a detectar un deep fake en imatges i vídeos són: inconsistències en la il·luminació i ombreig en la imatge o en un frame, alteració de la resolució en algunes zones i presència d'imperfeccions o incorreccions visibles.

SENSITY

sensity.ai/deepfakes-detection

DEEPWARE SCANNER

github.com/deepware

a la imatge que es vol analitzar i després fer clic en l'opció "*check fake profile picture*". En aquest moment apareixerà una notificació a la cantonada superior dreta que indica la probabilitat de què es tracti d'una imatge falsa.

FAKE PROFILE DETECTOR
e.digitall.org.es/fake-profile

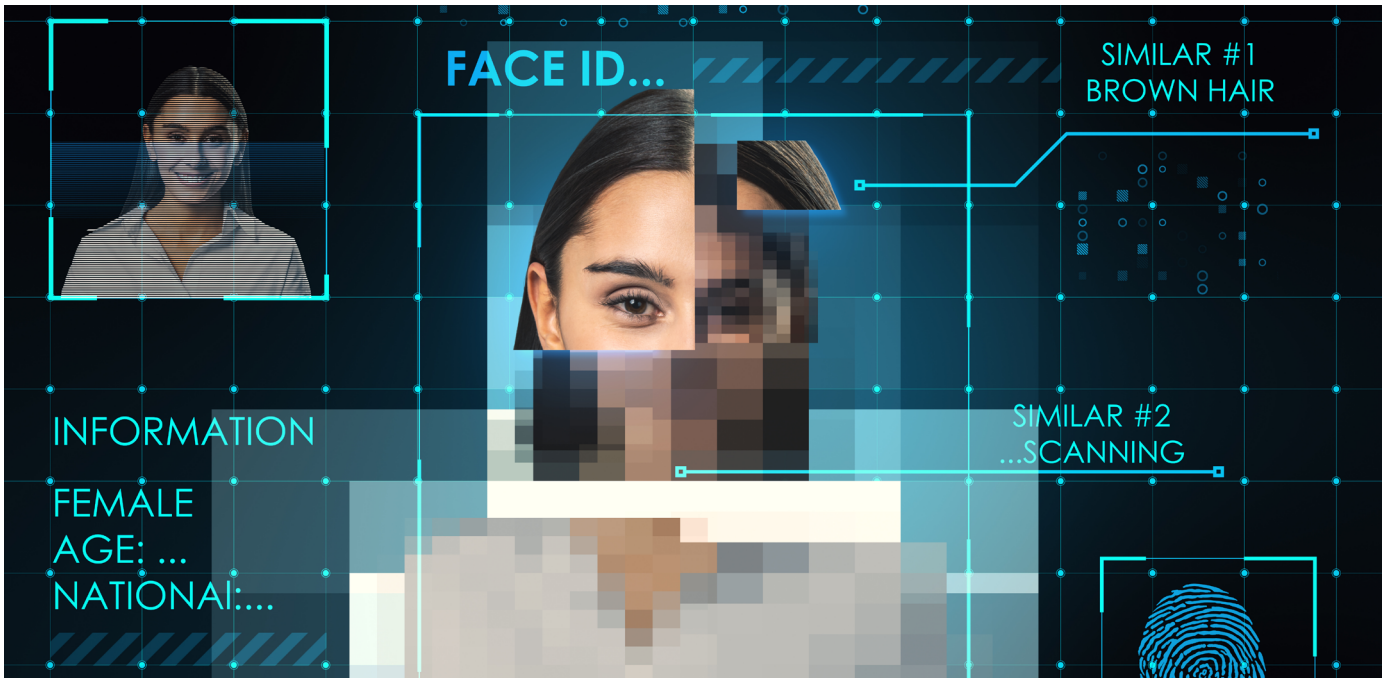
En resum, aquesta secció recull una sèrie de recomanacions pràctiques per autenticar contingut digital a partir de la inspecció visual, així com un conjunt d'eines pròpies de l'anàlisi forense informàtic i una altra tecnologia antihipertrucatge, que permeti a qualsevol usuari autenticar i verificar l'originalitat d'un contingut digital.

Saber-ne més

Una altra eina d'anàlisi forense és ***TinEye*** (*tineye.com*).

Saber-ne més

Programari de detecció d'hipertrucatge: *e.digitall.org.es/deepfake*



DigitAll

Creació de
continguts digitals

3.3
DRETS D'AUTOR I LLICÈNCIES DE PROPIETAT INTEL·LECTUAL

Creació de continguts digitals

Nivell C1 3.3 Drets d'autor i llicències de propietat intel·lectual

Registrant el copyright i explotant una obra

Registrant el copyright i explotant una obra

El contingut d'aquest document ofereix coneixements sobre el mecanisme de registre de la propietat intel·lectual de manera telemàtica per a un cas concret en la Comunitat de Castella-la Manxa, encara que serà igual per a qualsevol altra comunitat a excepció d'Andalusia i la Comunitat de Madrid que tenen sistemes propis.

Per sol·licitar el registre de manera telemàtica, a través de ***la seu electrònica del Ministeri de Cultura i Esport*** (***cultura.sede.gob.es/procedimientos***), cal accedir al llistat de procediments i seleccionar la categoria "***Propietat Intel·lectual***", un cop dins caldrà anar a la subcategoria "***Registre de la Propietat Intel·lectual***" i seleccionar la "***Sol·licitud de primera inscripció en el Registre de la Propietat Intel·lectual***" (***e.digitall.org.es/primera-inscripcion***). A continuació, es descriurà el procés per al registre d'una Obra artística, científica, de "PRIMERA INSCRIPCIÓ".

⚠ ATENCIÓ

La Sol·licitud de "**PRIMERA INSCRIPCIÓ**" es fa quan es vol sol·licitar la inscripció d'una obra que NO ha estat inscrita amb anterioritat en el Registre de la Propietat Intel·lectual (IPR), en què s'inclouen els següents casos:

1 | Sol·licituds d'obres no inscrites presentades pels autors, productors, etc.
2 | Sol·licituds d'obres no inscrites, els drets de les quals han estat transmesos inter vivos pels autors, productors, a un tercer.
3 | Sol·licituds d'obres no inscrites, els drets de les quals han estat transmesos mortis causa pels autors, productors, als seus hereus.

Per iniciar aquest procediment de sol·licitud de primera inscripció en el Registre de la Propietat Intel·lectual, una vegada en la web abans indicada, l'interessat haurà de fer clic al botó "Registrar" que apareix en la pàgina. Com el procediment pertany a diversos àmbits, posteriorment cal seleccionar la comunitat autònoma per al qual se sol·licita l'accés, en aquest cas particular se selecciona la comunitat autònoma de Castilla-la Manxa.

⚠ ATENCIÓ

En accedir s'informa sobre els requisits generals i tècnics necessaris per poder realitzar el registre. S'ha de triar el mètode d'identificació que s'utilitzarà per signar la sol·licitud que podrà ser mitjançant **SIGNATURA BÀSICA** (mitjançant Cl@ve) o **SIGNATURA AMB CERTIFICAT** (mitjançant Autofirm@, per la qual cosa haurà de tenir instal·lada aquesta aplicació).

Una vegada triat el mètode d'identificació, s'obre la finestra amb el formulari en línia per al registre, que contindrà quatre pestanyes: dades del sol·licitant, obra, Autors i documentació addicional. La pestanya dades de contacte del sol·licitant, ja conté algunes dades, no obstant això, caldrà completar "altres dades del sol·licitant", com són el sexe i la direcció. Per al registre s'han de completar dades en les altres pestanyes amb les dades de l'"obra", els "autors" de la mateixa i "documentació addicional", tal com es mostra en les següents figures:

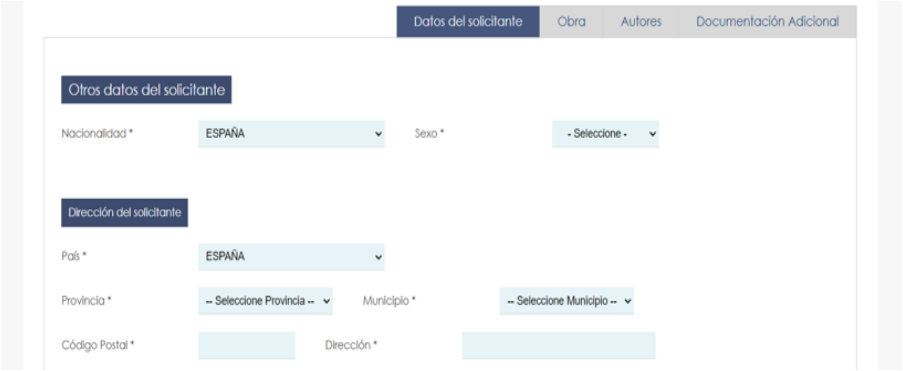


Figura 1. Pestanya: "Dades del sol·licitant".

Figura 2. Pestanya: "Obra".

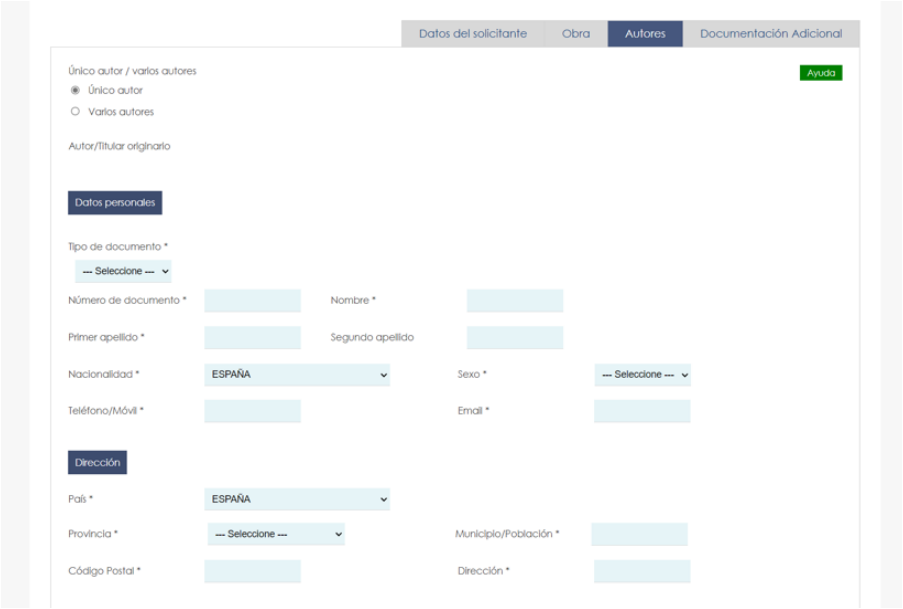


Figura 3. Pestanya: "Autors".

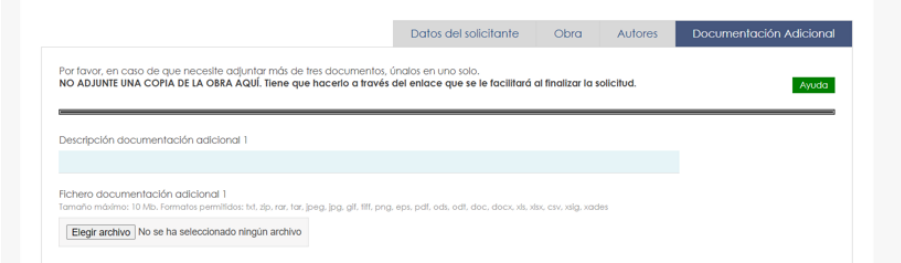


Figura 4. Pestanya: "Documentació addicional".

Una vegada emplenades totes les dades, al peu de la pàgina es troba la informació sobre la Taxa del Registre de la Propietat Intel·lectual, s'haurà d'indicar l'"Opció de pagament" seleccionada, que pot ser adjuntant el justificant de pagament que prèviament s'ha efectuat en alguna entitat bancària o per pagament telemàtic. En aquest darrer cas es podrà seleccionar l'entitat bancària en un desplegable i després el mètode de pagament que podrà ser a través d'un compte o targeta de crèdit. Una vegada introduïdes totes les dades requerides per al registre, per fer-ho efectiu s'haurà de fer clic en el botó "Enviar", que es troba en la part inferior del formulari.

Si tot és correcte, apareixerà una finestra emergent en la qual se li sol·licitarà a l'interessat que ferm la sol·licitud abans d'enviar-la. Si no indicarà els errors trobats sol·licitant que es corregeixin.

Una vegada signada la sol·licitud, li apareixerà una pantalla en la qual se li indica que la sol·licitud ha estat registrada amb èxit i se li mostren totes les dades introduïdes en la sol·licitud, en què s'inclou també el número de l'expedient que s'ha creat, el número de registre i la data. Es podrà prémer descarregar el justificant del registre de la sol·licitud en format PDF i també un document amb les dades de la Sol·licitud en format PDF.

Una vegada que l'interessat hagi registrat una obra, el sistema li oferirà una adreça web per procedir a l'aportació de la còpia de l'exemplar identificatiu.

Els documents que hauran d'adjuntar-se variaran segons la tipologia de l'obra que han de complir els requisits de tipus i grandària de fitxer permesos.

Saber-ne més

El document complet amb la descripció detallada de com cal presentar una sol·licitud telemàtica de primera inscripció es pot descarregar des de: *e.digitall.org.es/manual-primera-inscripcion*

DigitAll

Creació de
continguts digitals

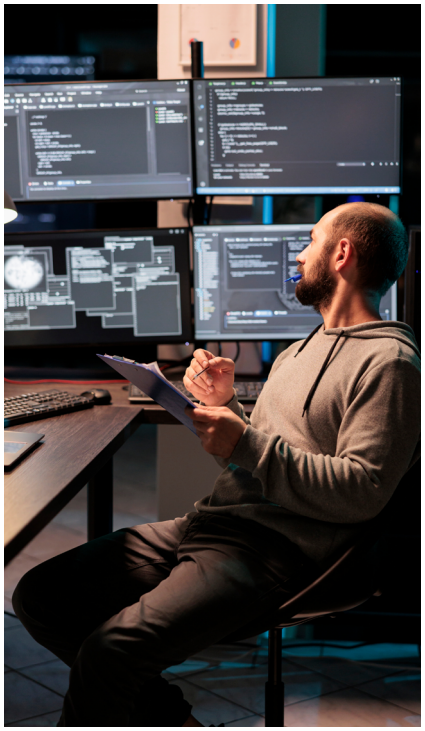
3.4
PROGRAMACIÓ

Creació de
continguts digitals

Nivell C1 3.4 Programació

Paradigmes de programació. Visió general

Paradigmes de programació. Visió general



Al llarg de la història de la programació, la manera de desenvolupar els programes ha anat variant en funció de la necessitat de resoldre problemes cada vegada més complexos i diversos. A l'enfocament o estil que determina com abordar el disseny i l'escriptura d'un programa se'l denomina paradigma de *programació*, i consisteix en un conjunt de regles, tècniques i principis que guien tot el procés. Els paradigmes de programació estan estretament relacionats amb els llenguatges de programació, ja que cada llenguatge de programació està dissenyat per suportar un o més paradigmes de programació. De fet, alguns llenguatges de programació són específics d'un sol paradigma, mentre que uns altres són multiparadigma, podent suportar diversos paradigmes diferents.

En l'actualitat, existeixen diferents paradigmes i cadascun posseeix característiques específiques que ho diferencien de la resta, relacionats amb la manera com es representen les dades i es manegen les operacions i el flux de control. Aquesta varietat és fruit de l'evolució que s'ha produït des del considerat com el paradigma més antic, el de la **programació imperativa**, l'origen de la qual es remunta al naixement dels primers llenguatges de programació, a la fi dels anys 50 del segle XX. La idea sobre la qual es fonamenta aquest paradigma és la que un programa és una seqüència d'instruccions que s'executen en un ordre específic. Així, sota aquest enfocament, els programes consisteixen en la descripció precisa i detallada de la seqüència de passos que cal fer per resoldre un problema; és a dir, el focus es posa en el "com" es resol el problema. Llenguatges típics, encara que alguns ja en desús, són Fortran, C, Pascal, Basic o Cobol.

A la fi de la dècada de 1960, els programes s'havien fet més complexos i va sorgir la necessitat de millorar la seva claredat, qualitat i temps de desenvolupament, la qual cosa va derivar en el paradigma de la programació estructurada. El principi en el qual es basa és el que qualsevol programa s'ha d'escriure fent ús exclusivament de les tres estructures de control bàsiques (seqüència, selecció i iteració) i de mòduls o procediments. Ada, Algol o Modula-2 són llenguatges de

programació estructurada. D'altra banda, el llenguatge C és principalment un llenguatge de programació imperatiu, però també té algunes característiques que el fan adequat per a la programació estructurada.

De manera gairebé paral·lela, es desenvolupa el paradigma de la **programació declarativa**, segons el qual, al contrari que en el paradigma imperatiu, els programes s'escriuen especificant "què" és el que ha de fer en comptes de "com" fer-ho. Per això, en lloc d'instruccions, s'utilitzen regles i propietats. Alguns llenguatges de programació moderns, com SQL per a bases de dades o HTML per al disseny de pàgines web, es basen en aquest paradigma. Dins d'aquest enfocament, existeixen dos paradigmes importants:

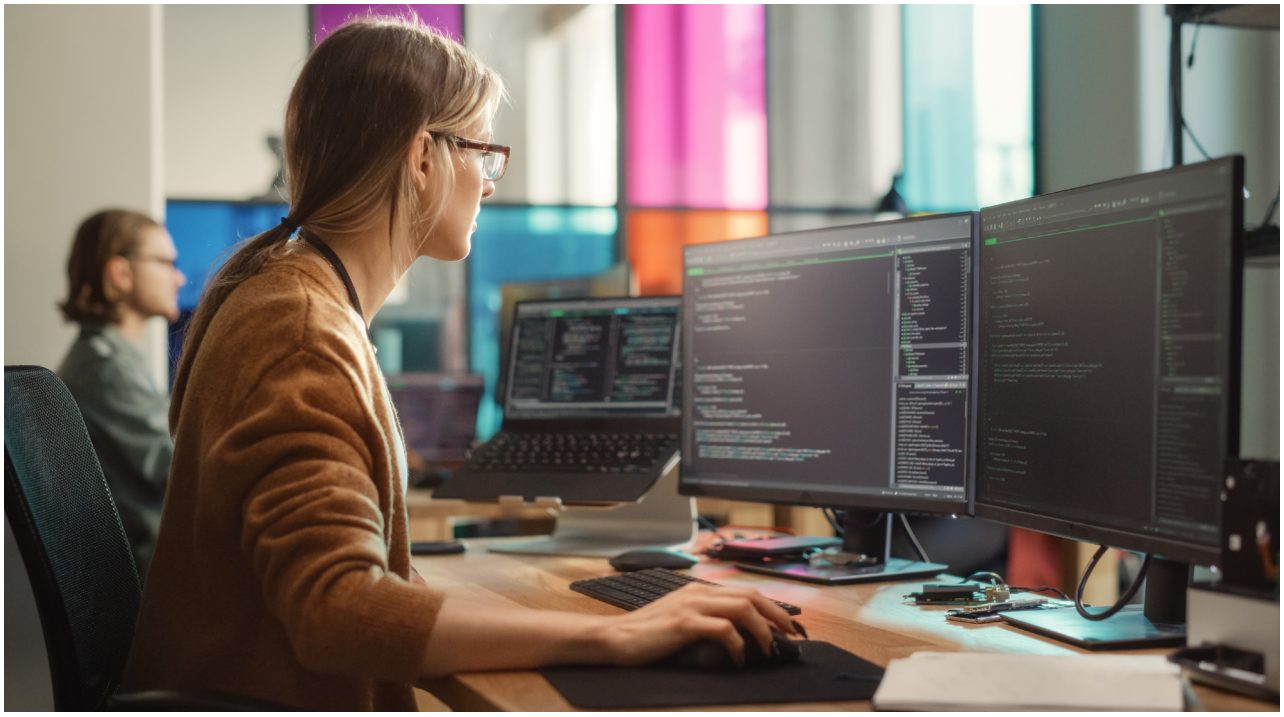
- ***Programació lògica,*** segons el qual el programa es construeix a partir de declaracions lògiques. El programa estableix una sèrie de fets i regles, i utilitza la lògica matemàtica per deduir conclusions. El llenguatge de programació Prolog, que es va desenvolupar en la dècada de 1970, va ser un dels primers llenguatges a implementar aquest paradigma.
- ***Programació funcional***, que es basa en l'ús exclusivament de funcions matemàtiques i en l'avaluació d'expressions. Aquest enfocament no va guanyar popularitat fins a la dècada dels 80 del segle XX. Lisp, Haskell i Scheme són llenguatges funcionals emprats bàsicament per a la recerca i l'ensenyament. Malgrat que s'ha emprat fonamentalment en la comunitat acadèmica, en els darrers anys ha guanyat popularitat en la indústria del programari, gràcies a la seva capacitat per manejar de manera eficient grans quantitats de dades i processos concurrents. Llenguatges com Clojure, Scala i F# són exemples de llenguatges de programació funcional que s'utilitzen en la indústria.

Una dècada més tard, durant els anys 90 del segle XX, es va popularitzar el paradigma de la **programació orientada a objectes (POO)** com a manera de representar la informació de manera més semblant a com ocorre en la vida real, així com per permetre la reutilització de codi. Sota aquest enfocament, un programa es considera una col·lecció d'objectes que interactuen entre si; fins i tot el mateix programa seria un

objecte. Alguns dels llenguatges de programació orientada a objectes més populars són Java, Python i C++. Aquests llenguatges moderns i multiparadigma permeten, a més, la programació imperativa i estructurada.

Un altre paradigma que és àmpliament utilitzat en l'actualitat per al desenvolupament d'aplicacions web o mòbils és el de la **programació orientada a esdeveniments**, a pesar que va sorgir a principis de la dècada de 1980. Neix de la necessitat de manejar interaccions de manera asíncrona en sistemes distribuïts i en temps real, com a sistemes de control industrial i de telecomunicacions. No obstant això, ha estat el desenvolupament de les interfícies gràfiques d'usuari el que li ha fet guanyar popularitat. Els llenguatges de programació **Java** i **C++** suporten aquest tipus de programació, encara que guanyen terreny altres llenguatges més actuals com **JavaScript** o **C#**.

Tots aquests paradigmes s'enquadren dins de la programació clàssica. En l'actualitat, comença a estendre's un paradigma completament distint, el de la **programació quàntica**. Aquest es basa a fer ús dels principis de la mecànica quàntica per desenvolupar programes, en els quals les operacions poden superposar-se i entrellaçar-se, cosa que permet fer diverses en paral·lel. Un exemple de llenguatge és Q#.



Creació de
continguts digitals

Nivell C1 3.4 Programació

Entorns de desenvolupament integrat (IDE). Visió general

Entorns de desenvolupament integrat (IDE). Visió general

Introducció

Els entorns de desenvolupament integrats, comunament coneguts com a IDE per la seva nomenclatura en anglès (*Integrated Development Environments*), són eines fonamentals per a programadors i desenvolupadors. Aquests entorns proporcionen un conjunt d'eines i funcionalitats integrades que simplifiquen i milloren el procés de desenvolupament de programari.

En aquest document, analitzarem les característiques principals dels IDE, així com els avantatges i inconvenients associats amb el seu ús. A més, compararem el seu enfocament amb el de l'intèrpret interactiu de Python.

Abans de continuar, és aconsellable que consultis el vídeo.

INTRODUCCIÓ ALS ENTORNS DE TREBALL DE DESENVOLUPAMENT

Aquest vídeo conté informació elemental sobre els conceptes d'IDE i les seves característiques principals.
e.digitall.org.es/A3C34B1V03

Què és un IDE?

Un IDE és un entorn de programació que agrupa diferents eines enfocades a la creació de programari. Dins de les quals, cal destacar les següents:

1 | Editor de codi
Proporciona una interfície per a escriure i editar codi font. Es tracta d'una potent eina que permet tant una escriptura més àgil com la reducció d'errors. Algunes de les característiques que inclou són el ressaltat de sintaxi, autocompletat i suggeriments contextuals.

2 | Depurador
La depuració de codi permet identificar errors i solucionar-los una vegada que han estat localitzats. Els IDE ofereixen al desenvolupador diverses opcions de depuració, com la inserció de punts de ruptura o la inspecció de variables.

3 | Navegació en el codi
Facilita una navegació ràpida i una cerca senzilla i eficient de determinats elements dins del codi. Permet realitzar cerques de funcions, classes i variables. A més, ofereix vistes jeràrquiques del projecte per millorar la comprensió de l'estructura del codi.

4 | Integració amb sistemes de control de versions
L'ús de sistemes de control de versions és una pràctica habitual entre els programadors. Per això, els IDE integren els sistemes de control de versions més estesos. Un sistema de control de versions permet fer un seguiment dels canvis realitzats en el codi al llarg del temps. D'aquesta manera, els desenvolupadors poden gestionar i controlar el versionat del seu codi sense haver de necessitar aplicacions externes.

5 | Proves
Les proves de programari són una etapa clau en la creació de qualsevol aplicació. Tan important com el disseny i desenvolupament de noves funcionalitats, és el disseny i desenvolupament de proves. Els IDE ofereixen eines i funcionalitats que faciliten el procés de prova i asseguren la qualitat del programari.

6 | Integració amb eines externes
Els IDE permeten la integració d'una àmplia varietat d'eines externes que complementen el procés de desenvolupament d'aplicacions. Aquestes eines abasten des del disseny i la planificació fins al desenvolupament, optimització i proves de programari.

Un entorn de desenvolupament integrat (IDE) és un programa informàtic enfocat a la creació de programari i que integra les eines més utilitzades pels desenvolupadors en una mateixa interfície gràfica.

Avantatges dels IDEs

Una vegada descrites les principals eines que integren un IDE, és fàcil identificar els avantatges que aporten als desenvolupadors de programari. L'avantatge més important i que impacta en altres aspectes és l'augment de la productivitat en totes les etapes del procés de desenvolupament. Els desenvolupadors compten amb una única interfície gràfica que reuneix les eines necessàries per cobrir totes les etapes del cicle de vida del programari, des de l'anàlisi i el disseny, fins a la codificació, les proves i el seu manteniment posterior.

Aquest avantatge es deu al fet que els desenvolupadors poden alternar de manera senzilla entre diferents tasques o fases del projecte, sense necessitat d'haver de canviar constantment d'aplicació. A més, els IDE són altament configurables, la qual cosa es tradueix en una adaptació a les necessitats i gustos particulars de cada desenvolupador.

Cal destacar que l'ús d'un IDE no només augmenta la productivitat en reduir el temps necessari per codificar programes, sinó que també contribueix a generar un codi de major confiança i més robust. Aquests dos avantatges venen derivats de l'ús de funcions com l'autocompletat, el ressaltat de sintaxi i la correcció d'errors en temps real, característiques que es troben disponibles en l'editor dels IDE.

Atès que la majoria dels desenvolupadors treballen en equip, és imprescindible establir uns estàndards comuns de treball. Els IDE proporcionen plantilles que poden seguir tots els membres de l'equip, facilitant el treball conjunt. A més, la integració d'eines de control de versions faciliten la compartició de codi de manera segura.

En resum, els IDE augmenten la productivitat dels desenvolupadors, redueixen el temps d'instal·lació i agilitzen les tasques de desenvolupament.

IDE vs. intèrpret de Python

En temes anteriors s'ha estudiat el concepte d'intèrpret interactiu, en particular l'intèrpret interactiu de Python. L'intèrpret interactiu de Python és un programa que llegeix instruccions escrites a Python, les avalua o processa i les executa. No obstant això, el concepte d'IDE és molt més complex i, a més, entre les seves eines compta amb intèrprets o compiladors.

INTÈRPRETS INTERACTIUS

Vídeo en el qual s'introdueix el concepte d'intèrpret interactiu, així com els avantatges que ofereix com a eina de programació. En concret, es particularitza amb l'intèrpret interactiu de Python, en què s'inclouen les nocions bàsiques d'instal·lació i d'execució.

e.digitall.org.es/A3C34B2V02

Els IDE presenten multitud d'avantatges, però a l'hora de desenvolupar un programa cal avaluar els avantatges i els principals inconvenients. Un és la complexitat de la seva corba d'aprenentatge, en concret per a aquells usuaris poc experts. A més, no només cal tenir en compte l'experiència del desenvolupador, sinó també el tipus de programa que es vol desenvolupar.

A continuació, es mostren una taula en la qual es compara l'ús, característiques i corba d'aprenentatge dels IDE enfront de l'intèrpret interactiu de Python.

	IDEs	Intèrpret de Python
Ús	Recomanat en projectes complexos.	Recomanat per executar proves ràpides i desenvolupaments senzills.
Característiques	Editor de codi amb ressaltat de sintaxi i auto-completat. - Integració amb eines externes i llibreries. - Major productivitat i agilitat en el desenvolupament de programari.	- Editor senzill. - Accés a la biblioteca estàndard de Python. - Fàcil accés i execució ràpida de codi Python
Corba d'aprenentatge	Complexa.	Senzilla.

En resum, l'elecció entre utilitzar un IDE o una altra eina, com l'intèrpret interactiu de Python, depèn de diversos factors com la complexitat del projecte, les necessitats específiques i l'experiència del desenvolupador. Els IDE són més adequats per a projectes complexos que requereixen característiques avançades, mentre que l'intèrpret interactiu de Python és més adequat per a desenvolupaments més simples i proves ràpides.

NOTA

Python inclou en la seva instal·lació bàsica un IDE bàsic anomenat IDLE (*Integrated Development and Learning Environment*). Ofereix característiques bàsiques d'un IDE, com un editor de codi amb ressaltat de sintaxi, autocompletat, cerca de text, i capacitat per executar codi de manera interactiva. Permet a més crear i executar scripts de Python, depurar codi pas a pas, accedir a la documentació integrada i executar proves unitàries.

Saber-ne més

A continuació, es llisten alguns dels IDE més populars entre els programadors de Python:

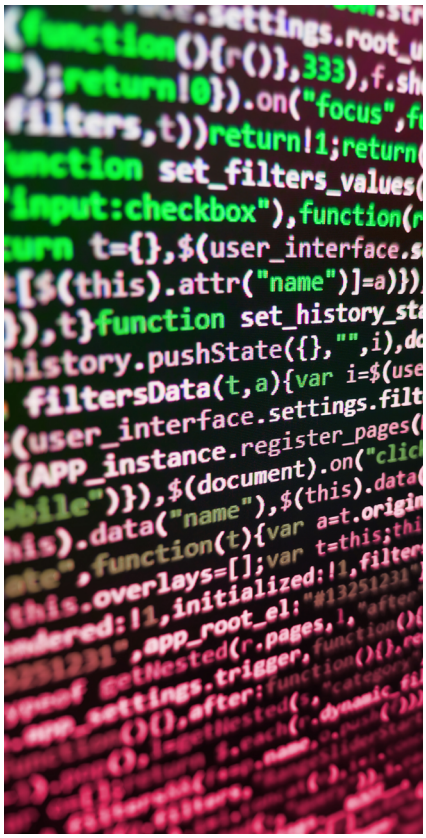
- **PyCharm:** *jetbrains.com/pycharm*
- **Spyder:** *spyder-ide.org*
- **Pydev:** *pydev.org*

Creació de
continguts digitals

Nivell C1 3.4 Programació

Excepcions. Què són i per a què serveixen?

Excepcions. Què són i per a què serveixen?



Amb caràcter general, una excepció es pot entendre com un esdeveniment que succeeix durant l'execució d'un programa i que altera el seu flux d'execució predefinit. En aquest sentit, l'ocurrència d'una excepció és una cosa no desitjada, ja que s'allunya del comportament esperat d'un programa. No obstant això, és possible capturar i manejar les excepcions que puguin ocórrer en temps d'execució, és a dir, quan un programa s'està executant, amb l'objectiu de recuperar la normalitat i continuar amb l'execució d'aquest.

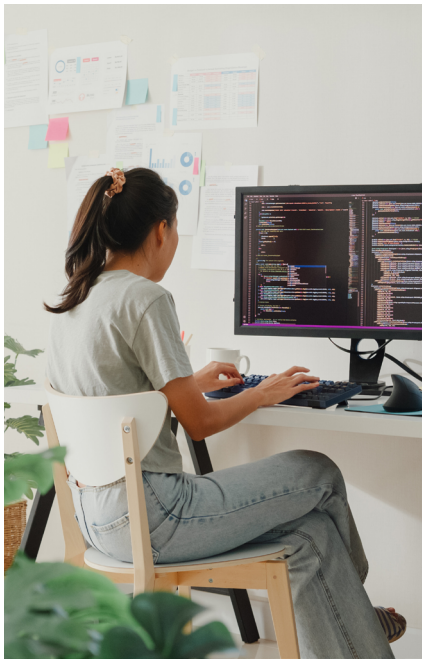
La majoria dels llenguatges de programació moderns, com Python, inclouen suport natiu per a la definició, captura i gestió d'excepcions. Això vol dir que el mateix llenguatge ofereix instruccions específiques per tractar excepcions, facilitant així el desenvolupament de programes robustos davant errors en temps d'execució. Aquest és la principal comesa de la gestió d'excepcions.

D'altra banda, és possible classificar les excepcions en: i) excepcions contemplades pel llenguatge de programació i ii) excepcions definides per l'usuari. Un exemple clàssic d'excepció contemplada pel llenguatge apareixeria quan una instrucció tracta de fer una divisió per zero. A Python, està excepció està definida per *ZeroDivisioError*. Un exemple d'excepció definida per l'usuari podria preveure l'intent d'emmagatzematge d'un número de telèfon que no tengui exactament 9 dígits. Respecte a aquest darrer exemple, l'usuari seria responsable de la definició d'aquest nou tipus d'excepció, de la mateixa manera que ocorreria amb la definició d'una nova estructura de dades.

⚠ ATENCIÓ

Quan es manegen excepcions, existeixen tres tipus d'operacions representatives: llançament d'excepcions, captura d'excepcions i maneig d'excepcions. El llançament es refereix a la capacitat del programador per disparar una excepció, que típicament serà recollida i manejada d'altra banda del programa. La captura implica la detecció explícita d'una instrucció a escala de codi. Finalment, el maneig està vinculat al codi font definit per tractar l'excepció prèviament capturada.

Exemple d'ús d'excepcions a Python



En aquesta secció es mostra un senzill exemple d'ús d'excepcions a Python. L'objectiu no és el d'aprofundir sobre com es defineixen i manegen excepcions a Python. Al contrari, simplement es pretén exemplificar el concepte d'excepció, introduït anteriorment, a través d'aquest llenguatge.

En aquest sentit, el següent fragment de codi mostra com es pot obtenir un número per teclat per a, posteriorment, emmagatzemar-lo en una variable denominada valor. Després, es mostra el contingut d'aquesta variable.

```
>>> valor = int(input("Introdueix un número..."))
Introdueix un número...7
>>> valor
7
```

Desafortunadament, aquest codi fallarà quan l'usuari introdueixi, per exemple, una cadena de text. En aquest cas, l'intèrpret de Python estarà esperant un valor (a causa de la conversió a sencer feta), per la qual cosa llançarà l'excepció ValueError.

```
>>> valor = int(input("Introdueix un número..."))
Introdueix un número...Ciutat Real
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'Ciutat Real'
```

A Python, com descobriràs amb detall més endavant, és possible embolicar un segment de codi per capturar les potencials excepcions que puguin succeir. Per això, pots utilitzar els blocs try-except. El següent fragment de codi captura i gestiona el possible error de l'usuari a l'hora d'introduir per teclat, en el nostre exemple anterior, alguna cosa que no sigui un valor sencer.

```
>>> try:
...     valor = int(input("Introdueix un número..."))
... except ValueError:
...     print("Has d'introduir un número")
...
Introdueix un número...Madrid
Has d'introduir un número
```

Creació de
continguts digitals

Nivell C1 3.4 Programació

Processament d'arxius a Python

Processament d'arxius a Python

Com s'ha vist anteriorment, Python incorpora una sèrie de funcions per a obrir, llegir, escriure i tancar arxius. Aquestes funcions es poden aplicar a qualsevol mena d'arxiu, encara que en aquesta guia es donaran indicacions per treballar concretament amb arxius de text pla.

Gestió bàsica d'errors

La gestió d'errors és una part crucial de qualsevol programa. A Python, els errors es gestionen utilitzant blocs **try/except**.

Si s'intenta obrir un arxiu que no existeix, Python llançarà un error FileNotFoundError. Aquest error pot ser capturat i gestionat emprant un bloc `try/except`, perquè l'execució del programa no finalitzi abruptament. Per exemple:

```
# Exemple de gestió derrors
try:
   with open("arxiu_inexistent.txt", "r") as arxiu:
      print(arxiu.read())
except FileNotFoundError:
   print("L'arxiu no ha estat trobat.")
```

En aquest cas, si l'arxiu no es troba, es mostrarà el missatge «L'arxiu no va ser trobat.» en pantalla, en lloc d'acabar el programa amb un error i continuant l'execució del programa en cas que hi hagués més sentències a continuació del bloc.

Cerca d'una cadena de text en el contingut d'un arxiu

Una tasca comuna en processar arxius és cercar una cadena de text específica dins del contingut de l'arxiu. A continuació, es presenta un exemple de com fer això, en què s'inclou el maneig de possibles errors que puguin sorgir:

```
# Exemple de cerca de text en un fitxer
nom_arxiu = "arxiu.txt"
cadena_cerca = "text a cercar"
try:
    with open(nom_arxiu, "r") as arxiu:
        contingut = arxiu.read()
        if cadena_cerca in contingut:
            print("La cadena de cerca va ser trobada a l'arxiu.")
        else:
            print("La cadena de cerca no s'ha trobat al fitxer.")
except FileNotFoundError:
    print(f"El fitxer {nom_arxiu} no s'ha trobat.")
```

En aquest codi, s'obre l'arxiu indicat per la variable `nom_arxiu`, es llegeix el seu contingut i, posteriorment, es comprova si la cadena de cerca està en el contingut de l'arxiu utilitzant l'operador `in`. Si la cadena de cerca està en l'arxiu, s'imprimeix un missatge indicant-lo. Si no hi està, s'imprimeix un missatge diferent. Si l'arxiu no es troba, s'imprimeix un missatge d'error.

Aquest exemple funciona correctament per a arxius que tenguin poc contingut. En cas que tenguem arxius amb una enorme quantitat de contingut (de l'ordre de gigabytes), aquesta tècnica probablement no funcionarà, ja que el programa no podrà carregar l'arxiu al complet en la memòria del sistema per a poder fer la cerca.

Per resoldre això, una possible solució seria llegir el contingut de l'arxiu línia a línia, de tal manera que només es carregui en memòria, com a molt, una línia de l'arxiu alhora. Per exemple:

```
# Exemple de cerca de text en un fitxer (versió millorada)
nom_arxiu = "arxiu.txt"
cadena_cerca = "text a cercar"
try:
    with open(nom_arxiu, "r") as arxiu:
        linia = arxiu.readline()
        while linia:
            if cadena_cerca in linia:
                print("La cadena de búsqueda fue encontrada en el archivo.")
                break
            linia = arxiu.readline()
        else:
            print("La cadena de cerca no s'ha trobat al fitxer.")
except FileNotFoundError:
    print(f"L'arxiu {nom_arxiu} no ha estat trobat.")
```

En aquest cas, s'obre un arxiu i es llegeix línia a línia. Si la cadena de cerca es troba en alguna línia de l'arxiu, s'imprimeix un missatge indicant-lo i s'interromp el cicle, de tal manera que no calgui continuar processant l'arxiu. Si no està, i s'han llegit totes les línies, s'imprimeix un missatge diferent. Si l'arxiu no es troba, s'imprimeix un missatge d'error.

Com s'ha comentat, aquesta versió del codi pot ser més eficient en termes de memòria quan es treballa amb arxius molt grans, ja que només manté una línia de l'arxiu en memòria alhora, en lloc del contingut complet de l'arxiu.

Conclusió

El processament d'arxius pot ser utilitzat per a una àmplia varietat de tasques, des de l'anàlisi de dades fins a l'automatització de tasques. Prèviament, s'han introduït tècniques per a obrir i llegir arxius, manejar errors, i cercar cadenes de text en el contingut d'un arxiu.

Saber-ne més

Per saber-ne més sobre la gestió d'errors a Python, pots consultar la documentació oficial: *e.digitall.org.es/excepciones*

Creació de
continguts digitals

Nivell C1 3.4 Programació

Programació orientada a objectes. Principis i conceptes fonamentals

Programació orientada a objectes. Principis i conceptes fonamentals

A l'hora de programar, els llenguatges de programació poden seguir diferents filosofies o paradigmes de programació, segons la realitat que pretenen modelar i el tipus de problemes que pretenen solucionar.

La realitat que ens envolta està repleta d'objectes com cadires, taules, cotxes, entre altres, que poden plantejar problemes que poden solucionar-se mitjançant programació. Amb la finalitat de representar aquesta realitat, el paradigma de **Programació Orientada a Objectes (POO)** posa l'accent principalment en el disseny i la manipulació d'objectes com a elements fonamentals per a dissenyar la solució per a qualsevol programa. A continuació, es descriuran els principis i conceptes fonamentals d'aquest paradigma.

Conceptes fonamentals

A hora de modelar la solució a un problema de la vida real, és necessari treballar amb objectes tan diversos com una cadira o una taula que poden formar part del problema. Per tant, és necessari descriure formalment aquests objectes, però no totes les seves característiques, sinó les que són més importants per al programa.

Abstracció

L'abstracció és un dels pilars de la POO. Consisteix en un procés de reconeixement d'aquelles característiques que són fonamentals per a la realitat que es vol modelar. Ha de permetre obviar aquells detalls que són irrellevants per a un programa, centrant-se únicament en el nombre mínim de detalls que s'han d'implementar.

Suposem que es vol implementar un programa en el qual intervinguin persones. El primer que s'ha de fer és modelar el concepte de **Persona**. Els elements bàsics que poden definir a aquesta persona poden ser de dos tipus: **atributs** i **accions**. Els **atributs** són les característiques o variables que la defineixen, com podrien ser el seu nom i edat. Les **accions** serien els mètodes que actuarien sobre aquests atributs.

Tipus	Persona
Atributs	nom edat
Accions	demanarNom Aniversari

Una persona pot tenir més característiques com el sexe o el DNI. No obstant això, fent un procés d'abstracció, és necessari determinar quins són els atributs i accions mínimes que són necessaris per al programa a implementar. En aquesta figura podria veure's l'esquema general d'un tipus Persona. Considerant aquesta definició o esquema general del que seria una Persona en un programa, en **POO** es correspondria amb el concepte de **classe**.

Concepte de classe

Fins ara, per a la gestió de dades a Python, s'han emprat tipus de dades primitives com *int*, *float* o *string*. Aquestes permeten representar un tipus de dada senzilla com és el cas d'un sencer o una cadena de text. No obstant això, hi ha situacions en les quals és necessari modelar tipus de dades més complexes que es formen com a composició d'altres dades més simples. Per a això, es pot recórrer al concepte d'estructures de dades més complexes com a vectors o matrius. Aquestes normalment solen ser homogènies, és a dir, contenen el mateix tipus de dades, tots els valors són numèrics o tots són cadenes de text, per exemple. També, existeix la possibilitat que siguin heterogènies, és a dir, poden combinar-se valors sencers, amb valors reals o cadenes text, per exemple.

Si a més de treballar sobre diferents tipus de dades, s'afegeixen operacions que permetessin manipular aquestes dades, s'estaria davant la definició d'un tipus abstracte de dades. Per tant, el concepte de classe és un tipus d'abstracte de dades, és a dir, un nou tipus, a part dels bàsics ja coneguts com a sencers o cadenes de text.

Resumidament, una classe representa una entitat els **atributs** de la qual seran implementats mitjançant les variables d'aquesta **classe** i les accions que pugui dur a terme mitjançant un conjunt funcions. Pot considerar-se com un **Tipus abstracte de dades**, que inclou les dades o variables sobre els quals operar i les accions o mètodes que es poden fer sobre aquestes dades. Un exemple d'implementació a Python de la classe Persona vista anteriorment seria:

```
#CODI FONT DEL TIPUS PERSONA

class Persona(object):
   def __init__(self, nom, edat):
      self.nom = nom
      self.edat = edat
   def demanarNom(self):
      return self.nom
   def Aniversari (self):
      self.edat += 1
```

La funció demanarNom permet consultar l'atribut nom de cada objecte Persona implementat, mentre que la funció Aniversari permet actualitzar el valor de la variable edat de cada objecte Persona. D'aquesta manera s'aplica el principi d'encapsulació, un altre dels pilars de la POO. Les dades edad i nombre estan protegits, no s'hi accedeix directament, només a través de mètodes especialment dissenyats per a la seva consulta i manipulació.

A més, cal destacar el mètode __init__ que rep el nom de mètode constructor o mètode d'inicialització d'atributs (edat i nom) d'un objecte.

Concepte d'objecte

Així, una classe és un nou tipus de dades, igual que ja s'havia vist anteriorment el tipus *boolean* o *int*, que de manera abstracta representen un valor lògic o sencer. No obstant això, per poder operar sobre un valor concret, era necessària la definició d'una variable. En POO existeix el concepte *anàleg*, que és el concepte de referència a un objecte.

Raonant per analogia, la manera de crear una variable és donar-li un nom i un valor inicial. La manera de crear un objecte és la mateixa, i consisteix a donar-li un nom a la referència i assignar els valors de tots els seus atributs.

```
nomVariable = valorInicial
nomReferencia = valors inicials atributs
```

Com un objecte té diversos atributs, no hi haurà un únic valor inicial sinó diversos. Per aquesta raó, el mètode constructor és l'encarregat d'inicialitzar tots els atributs alhora.

```
pep = Persona(nombre = "Josep", edat = "20")  #inicialització d'un objecte
```

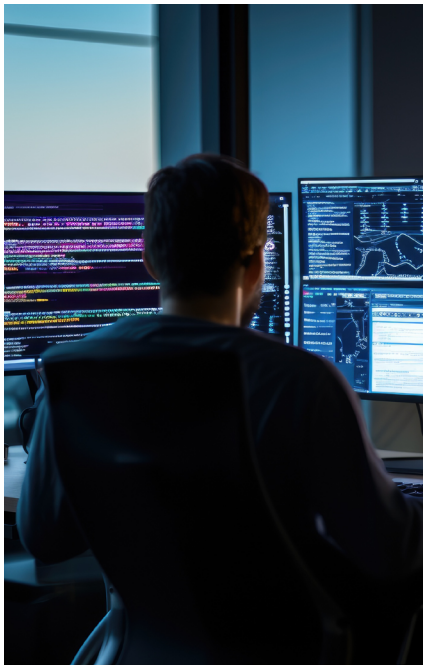
Una classe es defineix únicament una vegada, no obstant això, es crearan tants objectes, és a dir, tantes persones, com es requereixin en un programa.

Una de les característiques principals dels objectes és el seu estat. L'estat representa la situació actual dels seus atributs, és a dir, el valor de les seves variables. L'estat d'un objecte és dinàmic. Així, una persona avui pot tenir 20 anys, i demà 21, és a dir, el seu estat pot evolucionar amb el temps.

Per invocar qualsevol acció o mètode d'un objecte, a Python, s'utilitzarà la referència creada al costat de l'operador punt "**.**". Així, si es vol que una persona compleixi un any més o preguntar com es diu, simplement és necessari invocar els mètodes d'aquesta manera:

```
pep.aniversari()
print(pep.demanarNom())
```

Relacions entre objectes/classes



Les classes i objectes no són elements estancs que existeixen sense més ni més. Igual que en la vida real, els objectes interacció estableixen relacions. Existeixen diferents tipus de relacions que es pot caracteritzar per diferents tipus de predicats.

1 | Relacions d'associació

S'identifiquen per predicats del tipus: "**té un o diversos, etc.**"

Representen una relació estructural entre objectes, és a dir, una relació que perdura en el temps. Un exemple seria un cotxe té diverses rodes. El cotxe pot representar-se com una classe que pot tenir atributs marca i matrícula, i una roda pot representar-se com una classe amb atributs com a marca, pressió, i diàmetre.

2 | Relacions de dependència

S'identifiquen per predicats del tipus: "**usa/coneix un o diversos, etc**."

Representen una relació temporal entre objectes. Un exemple seria una persona empra un bolígraf. Al contrari que l'exemple

anterior, el cotxe sempre té rodes, no obstant això, una persona no sempre té un bolígraf amb si, simplement l'usa quan és necessari i perd la seva relació després d'emprar-lo.

3 | Relacions d'herència

L'herència és un altre dels pilars bàsics de la POO. Les relacions d'herència s'identifiquen per predicats del tipus: "**és un...**"

Representen relacions en les quals els objectes comparteixen característiques i comportaments, és a dir, mètodes i atributs, encara que també hi hagi unes certes diferències que els fan únics.

Un exemple seria les classes **animal**, **peix** i **gos**. El peix és un tipus d'animal i el gos és un tipus d'animal, per la qual cosa tindran coses en comú, per exemple, ambdós tenen una edat o se'ls pot posar un nom. No obstant això, hi ha característiques que no són comunes, com la manera de moure's, un res i l'altre camina.

L'herència és, a més, la base de **polimorfisme**, un altre dels pilars de la POO. Permet, per exemple, que una matriu de la classe Animal pugui contenir diversos objectes tant de la classe Peix o de la classe Gos.

Cadascuna d'aquestes relacions té la seva pròpia implementació en el llenguatge de programació en el qual es treballi.

Saber-ne més

Python.org. Definició de classes: *e.digitall.org.es/python-clases*

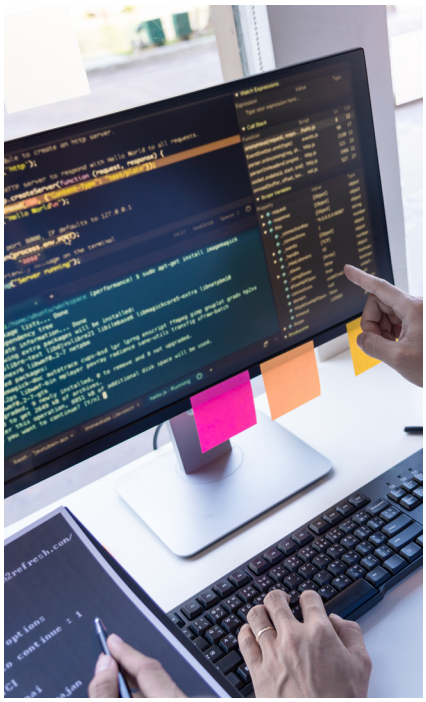
Microsoft Company. Què és la POO?: *e.digitall.org.es/poo*

Creació de
continguts digitals

Nivell C1 3.4 Programació

Proves de codi. Aspectes fonamentals

Proves de codi. Aspectes fonamentals



En el cicle de desenvolupament de programari, se segueixen diverses etapes que van des de l'anàlisi dels requisits fins al manteniment del sistema en funcionament. Una de les etapes més crítiques en aquest cicle és la de proves. Les proves de codi, o tests, són una part integral de qualsevol procés de desenvolupament de programari. Proporcionen una manera d'assegurar que el codi escrit funcioni com s'esperava i compleixi amb els requisits especificats.

Cicle de desenvolupament de programari

El cicle de desenvolupament de programari es pot dividir en les següents etapes:

1 | **Requisits:** es recullen i analitzen els requisits del sistema a desenvolupar.
2 | **Disseny:** es decideix l'arquitectura del sistema i es dissenyen els diferents components.
3 | **Desenvolupament:** s'implementa el sistema basant-se en els dissenys.
4 | **Proves**: es verifica que el sistema compleixi amb els requisits i es comporta correctament.
5 | **Integració:** es combinen tots els components del sistema i es verifiquen com un tot.
6 | **Manteniment:** es fa un seguiment del sistema en funcionament, corregint errors i adaptant-lo a noves necessitat.

Les proves es realitzen per identificar errors, fallades o discrepàncies entre el sistema construït i els requisits originals. No només impliquen trobar i arreglar errors en el codi, sinó també verificar que el sistema compleix amb els requisits i validar que és el que l'usuari necessita. Les proves són una manera de garantir la qualitat del programari i reduir la quantitat de problemes que es troben en l'etapa d'integració.

En les següents pàgines, explorarem l'enfocament de desenvolupament guiat per proves (*Test-Driven Development o TDD*), la seva relació amb les proves de codi i com es poden implementar a Python.

Desenvolupament Guiat per Proves (TDD)

El Desenvolupament Guiat per Proves (*Test-Driven Development*) és una metodologia de desenvolupament de programari que gira entorn de la repetició d'un cicle de desenvolupament molt curt: primer el desenvolupador escriu un cas de prova automatitzat que defineix una millora desitjada o una nova funció, després produeix el codi mínim necessari per passar aquesta prova i finalment refina el nou codi al nivell acceptable.

Enfocament TDD vs. tradicional

En l'enfocament tradicional, el desenvolupament de programari tendeix a seguir aquest ordre: i) s'escriu el codi, ii) es du a terme una prova per a verificar que funcioni correctament i iii) si es troben errors, es corregeixen i es torna a provar. Aquest procés continua fins que el codi passa totes les proves.

En contrast, TDD canvia completament aquest enfocament. Abans d'escriure qualsevol codi de producció, el desenvolupador primer escriu una prova per al nou codi. Inicialment, aquesta prova fallarà perquè el codi que està provant encara no existeix. Després, el desenvolupador escriu el codi mínim necessari perquè la prova passi. Finalment, es refina el codi, millorant-lo sense canviar el seu comportament.

TDD a Python

Python, com que és un llenguatge de programació d'alt nivell i de propòsit general, proporciona un conjunt d'eines que permeten implementar la metodologia TDD de manera efectiva.

Per exemple, suposem que es vol implementar una funció que sumi dos números. En un enfocament TDD, primer s'escriuria una prova per a aquesta funció:

```
import unittest

class TestSuma(unittest.TestCase):
    def test_suma(self):
        resultat = sumar(1, 2)
        self.assertEqual(resultat, 3)

if __name__ == '__main__':
    unittest.main()
```

En executar aquest codi, la prova fallarà perquè encara no s'ha definit la funció sum(). Després d'això, s'implementaria la funció per fer que la prova passi:

```
def sum(a, b):
    return a + b
```

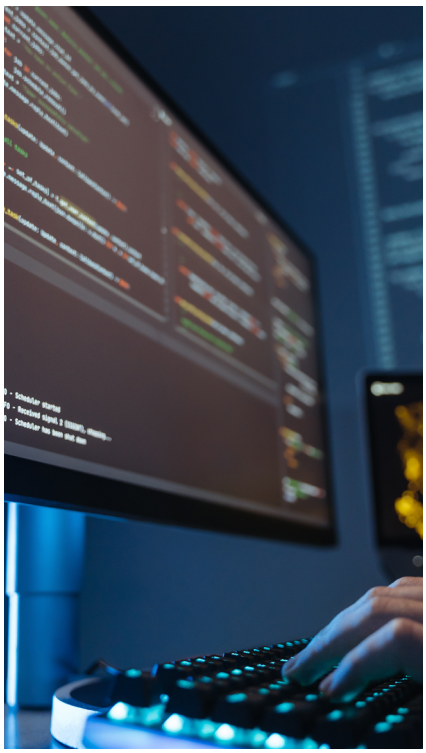
I es tornarien a executar les proves. Ara haurien de passar correctament, indicant que la nostra funció compleix amb l'expectativa que hem definit en la prova.

Conclusió

Les proves de codi són una part integral de qualsevol procés de desenvolupament de programari. Proporcionen una manera d'assegurar que el programari desenvolupat es comporta d'acord amb les expectatives i compleix amb els requisits definits. D'altra banda, el Desenvolupament Guiat per Proves (TDD) és una metodologia que posa les proves en el centre del desenvolupament de programari, promovent l'escriptura de codi d'alta qualitat que compleix amb les necessitats de l'usuari i minimitza l'aparició d'errors.

Python, amb el seu conjunt d'eines de prova i la seva sintaxi clara i concisa, és una excel·lent opció per implementar la metodologia TDD. L'enfocament d'escriure proves abans que el codi de producció ajuda a definir clarament les expectatives per al codi, facilita la detecció d'errors i promou l'escriptura de codi net i mantenible en el temps.

Les proves són una inversió que pot estalviar molt de temps i esforç en el futur, evitant l'aparició d'errors inesperats i facilitant la detecció i correcció d'aquests quan succeeixen. Adoptar un enfocament de desenvolupament centrat en les proves pot millorar la qualitat del programari i fer que el procés de desenvolupament sigui més eficient i efectiu.



Saber-ne més

Per saber-ne més sobre el desenvolupament guiat per proves a Python, pots consultar la documentació oficial en català dels següents mòduls:

- **unittest:** *e.digitall.org.es/unittest*
- **doctest:** *e.digitall.org.es/doctest*

DigitAll

Formació en
Competències
Digitals

https://www.digitall.org.es/

DigitAll

Coordinación General

Universidad de Castilla-La Mancha
Carlos González Morcillo
Francisco Parreño Torres

Coordinadores de área

Área 1. Búsqueda y gestión de información y datos

Universidad de Zaragoza
Francisco Javier Fabra Caro

Área 2. Comunicación y colaboración

Universidad de Sevilla
Francisco Javier Fabra Caro
Francisco de Asís Gómez Rodríguez
José Mariano González Romano
Juan Ramón Lacalle Remigio
Julio Cabero Almenara
María Ángeles Borrueco Rosa

Área 3. Creación de contenidos digitales

Universidad de Castilla-La Mancha
David Vallejo Fernández
Javier Alonso Albusac Jiménez
José Jesús Castro Sánchez

Área 4. Seguridad

Universidade da Coruña
Ana M. Peña Cabanas
José Antonio García Naya
Manuel García Torre

Área 5. Resolución de problemas

UNED
Jesús González Boticario

Coordinadores de nivel

Nivel A1

Universidad de Zaragoza
Ana Lucía Esteban Sánchez
Francisco Javier Fabra Caro

Nivel A2

Universidad de Córdoba
Juan Antonio Romero del Castillo
Sebastián Rubio García

Nivel B1

Universidad de Sevilla
Francisco de Asís Gómez Rodríguez
José Mariano González Romano
Juan Ramón Lacalle Remigio
Montserrat Argandoña Bertran

Nivel B2

Universidad de Castilla-La Mancha
María del Carmen Carrión Espinosa
Rafael Casado González
Víctor Manuel Ruiz Penichet

Nivel C1

UNED
Antonio Galisteo del Valle

Nivel C2

UNED
Antonio Galisteo del Valle

Maquetación

Universidad de Salamanca
Fernando De la Prieta Pintado
Pilar Vega Pérez
Sara Alejandra Labrador Martín

Creadores de contenido

Área 1. Búsqueda y gestión de información y datos

1.1 Navegar, buscar y filtrar datos, información y contenidos digitales

Universidad de Huelva
Ana Duarte Hueros (coord.)
Arantxa Vizcaíno Verdú
Carmen González Castillo
Dieter R. Fuentes Cancell
Elisabetta Brandi
José Antonio Alfonso Sánchez
José Ignacio Aguaded
Mónica Bonilla del Río
Odiel Estrada Molina
Tomás de J. Mateo Sanguino (coord.)

1.2 Evaluar datos, información y contenidos digitales

Universidad de Zaragoza
Ana Belén Martínez Martínez
Ana María López Torres
Francisco Javier Fabra Caro
José Antonio Simón Lázaro
Laura Bordonaba Plou
María Sol Arqued Ribes
Raquel Trillo Lado

1.3 Gestión de datos, información y contenidos digitales

Universidad de Zaragoza
Ana Belén Martínez Martínez
Francisco Javier Fabra Caro
Gregorio de Miguel Casado
Sergio Ilarri Artigas

Área 2. Comunicación y colaboración

2.1 Interactuar a través de tecnología digitales

Iseazy

2.2 Compartir a través de tecnologías digitales

Universidad de Sevilla
Alién García Hernández
Daniel Agüera García
Jonatan Castaño Muñoz
José Candón Mena
José Luis Guisado Lizar

2.3 Participación ciudadana a través de las tecnologías digitales

Universidad de Sevilla
Ana Mancera Rueda
Félix Biscarri Triviño
Francisco de Asís Gómez Rodríguez
Jorge Ruiz Morales
José Manuel Sánchez García
Juan Pablo Mora Gutiérrez
Manuel Ortigueira Sánchez
Raúl Gómez Bizcocho

2.4 Colaboración a través de las tecnologías digitales

Universidad de Sevilla
Belén Vega Márquez
David Vila Viñas
Francisco de Asís Gómez Rodríguez
Julio Barroso Osuna
María Puig Gutiérrez
Miguel Ángel Olivero González
Óscar Manuel Gallego Pérez
Paula Marcelo Martínez

2.5 Comportamiento en la red

Universidad de Sevilla
Ana Mancera Rueda
Eva Mateos Núñez
Juan Pablo Mora Gutiérrez
Óscar Manuel Gallego Pérez

2.6 Gestión de la identidad digital

Iseazy

Área 3. Creación de contenidos digitales

3.1 Desarrollo de contenidos

Universidad de Castilla-La Mancha
Carlos Alberto Castillo Sarmiento
Diego Cordero Contreras
Inmaculada Ballesteros Yáñez
José Ramón Rodríguez Rodríguez
Rubén Grande Muñoz

3.2 Integración y reelaboración de contenido digital

Universidad de Castilla-La Mancha
José Ángel Martín Baos
Julio Alberto López Gómez
Ricardo García Ródenas

3.3 Derechos de autor (copyright) y licencias de propiedad intelectual

Universidad de Castilla-La Mancha
Gabriela Raquel Gallicchio Platino
Gerardo Alain Marquet García

3.4 Programación

Universidad de Castilla-La Mancha
Carmen Lacave Rodero
David Vallejo Fernández
Javier Alonso Albusac Jiménez
Jesús Serrano Guerrero
Santiago Sánchez Sobrino
Vanesa Herrera Tirado

Área 4. Seguridad

4.1 Protección de dispositivos

Universidade da Coruña
Antonio Daniel López Rivas
José Manuel Vázquez Naya
Martiño Rivera Dourado
Rubén Pérez Jove

4.2 Protección de datos personales y privacidad

Universidad de Córdoba
Aida Gema de Haro García
Ezequiel Herruzo Gómez
Francisco José Madrid Cuevas
José Manuel Palomares Muñoz
Juan Antonio Romero del Castillo
Manuel Izquierdo Carrasco

4.3 Protección de la salud y del bienestar

Universidade da Coruña
Javier Pereira Loureiro
Laura Nieto Riveiro
Laura Rodríguez Gesto
Manuel Lagos Rodríguez
María Betania Groba González
María del Carmen Miranda Duro
Nereida María Canosa Domínguez
Patricia Concheiro Moscoso
Thais Pousada García

4.4 Protección medioambiental

Universidad de Córdoba
Alberto Membrillo del Pozo
Alicia Jurado López
Luis Sánchez Vázquez
María Victoria Gil Cerezo

Área 5. Resolución de problemas

5.1 Resolución de problemas técnicos

Iseazy

5.2 Identificación de necesidades y respuestas tecnológicas

Iseazy

5.3 Uso creativo de la tecnología digital

Iseazy

5.4 Identificar lagunas en las competencias digitales

Iseazy

Financiado por la Unión Europea NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia