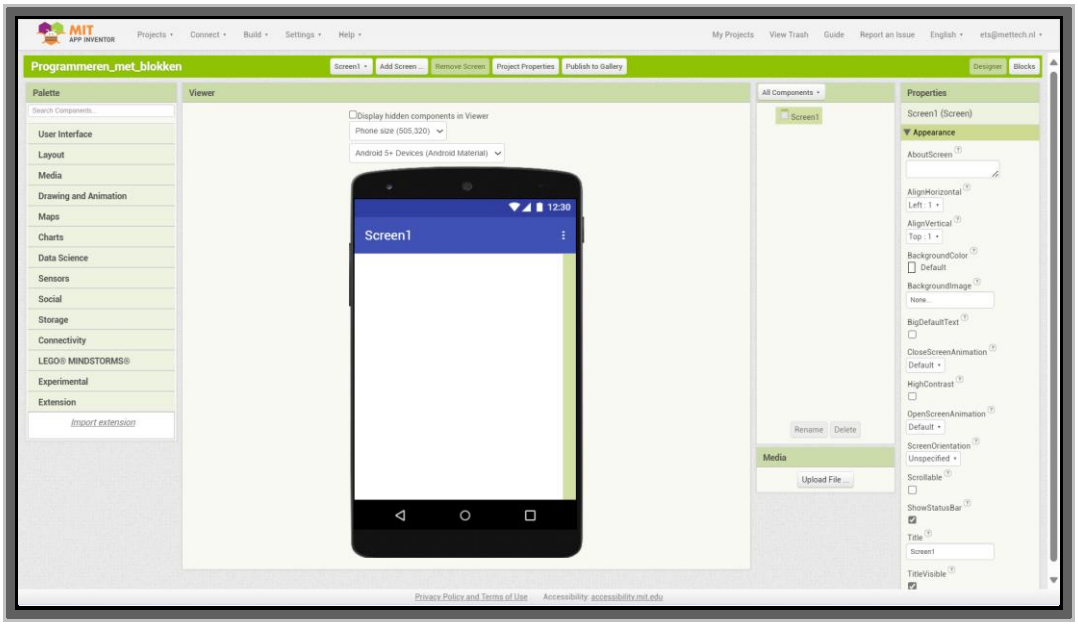
MIT App Inventor



MIT App Inventor, ingebouwde blokken

Ingebouwde blokken zijn beschikbaar, ongeacht welke componenten zich in uw project bevinden. Naast deze taalblokken heeft elk onderdeel in uw project zijn eigen set blokken die specifiek zijn voor zijn eigen gebeurtenissen, methoden en eigenschappen. Dit is een overzicht van alle ingebouwde blokken die beschikbaar zijn in de blokkeneditor.

Voorwoord

Omdat ik nogal eens aan het zoeken ben welke functies beschikbaar zijn en hoe ze toegepast moeten worden, heb ik deze handleiding gekopieerd van de site van MIT App Inventor. Vooral handig wanneer je niet met twee schermen werkt.

Speciale informatie voor scholen

MIT App Inventor hecht veel waarde aan de relatie met het onderwijs gemeenschap en is vereerd dat veel docenten de MIT-app hebben gevonden Inventor om een waardevol hulpmiddel te zijn voor klassikaal leren. MIT ook hecht veel waarde aan de privacy van zijn gebruikers, zoals uitgelegd in de MIT App Inventor Privacybeleid en gebruiksvoorwaarden.

Omdat veel staten nu wetten hebben aangenomen met betrekking tot het gebruik van online platforms in de klas, MIT-app Inventor is gevraagd om bepaalde "addenda" uit te voeren om service te verlenen contracten om het gebruik van MIT App Inventor voort te zetten voor gebruik in de klas.

MIT App Inventor is niet in staat om overeenkomsten uit te voeren van Dit soort omdat het geen contractuele afspraken met de school nakomt districten als onderdeel van deze dienst. In plaats daarvan is MIT App Inventor een Gratis, online tool beschikbaar voor iedereen die zich registreert voor een account.

Als u MIT App Inventor wilt gebruiken voor gebruik in de klas, hebben we een andere versie van het platform ontwikkeld die geen of persoonlijk identificeerbare informatie (PII) bij te houden. Dit versie is verkrijgbaar bij http://code.appinventor.mit.edu en kan geschikter zijn voor uw gebruik. Als u nog meer Vragen, neem dan contact met ons op via appinventor@mit.edu.

Ingangsdatum: 1 juli 2020

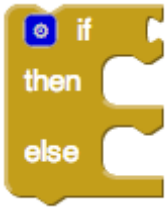
Het bovenstaande is automatisch vertaald!

Inhoud

Controleblokken

if & else if

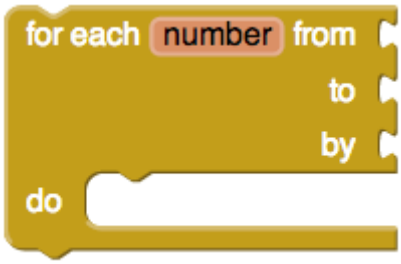
Test een bepaalde voorwaarde. Als de voorwaarde waar is, voert u de acties uit in een bepaalde reeks blokken; anders worden de blokkades genegeerd.



Test een bepaalde voorwaarde. Als de voorwaarde waar is, voert u de acties uit in de -dan-reeks blokken; anders voert de acties uit in de -else equence van blokken.

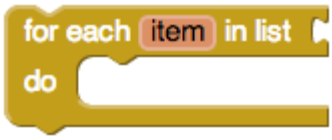
Test een bepaalde voorwaarde. Als het resultaat waar is, voert u de acties uit in de -dan-reeks blokken; anders test de instructie in de sectie -else if. Als het resultaat

voor elk getal van tot



Hiermee voert u de blokken in de sectie do uit voor elke numerieke waarde in het bereik dat begint bij en eindigt bij tot, waarbij elke keer wordt verhoogd met de waarde van by. Gebruik de naam van de variabele, , om naar de huidige waarde te verwijzen. U kunt de naam desgewenst in iets anders wijzigen.numbernumbernumber

voor elk item in de lijst



Voert de blokken uit in de doe-sectie voor elk item in de lijst. Gebruik de opgegeven variabelenaam, , om naar het huidige lijstitem te verwijzen. U kunt de naam desgewenst in iets anders wijzigen.itemitem

Terwijl

Test de -test-voorwaarde. Als dit waar is, voert u de actie uit die is gegeven in -do en test u opnieuw. Wanneer de test vals is, wordt de blokkering beëindigd en wordt de actie die in -do wordt gegeven niet meer uitgevoerd.

als dan anders

Test een bepaalde voorwaarde. Als de instructie waar is, voert u de acties uit in de dan-retourreeks blokken en retourneert u de dan-retourwaarde; anders voert u de acties uit in de else-return-reeks blokken en retourneert u de else-return-waarde. Dit blok is vergelijkbaar met de ternaire operator (?:) die in sommige talen voorkomt.

doen met resultaat

Soms moet u in een procedure of een ander codeblok iets doen en iets teruggeven, maar om verschillende redenen kunt u ervoor kiezen om dit blok te gebruiken in plaats van een nieuwe procedure te maken.

Evalueer maar negeer het resultaat

Biedt een "dummy-aansluiting" voor het plaatsen van een blok met een stekker aan de linkerkant op een plaats waar geen stopcontact is, zoals een van de opeenvolging van blokken in het doe-gedeelte van een procedure of een if-blok. Het blok waarin u past, wordt uitgevoerd, maar het geretourneerde resultaat wordt genegeerd. Dit kan handig zijn als u een procedure definieert die een resultaat retourneert, maar deze wilt aanroepen in een context die geen resultaat accepteert.

Open een ander scherm

Opent het scherm met de opgegeven naam.

De schermnaam moet een van de schermen zijn die met de ontwerpfunctie zijn gemaakt. De screenName moet worden geselecteerd in het vervolgkeuzemenu voor de naam van het verbonden scherm.

Als u een ander scherm opent, moet u het sluiten wanneer u terugkeert naar uw hoofdscherm om systeemgeheugen vrij te maken. Het niet sluiten van een scherm bij het verlaten ervan zal uiteindelijk leiden tot geheugenproblemen.

App-ontwikkelaars mogen Screen1 nooit sluiten of deze blokkering gebruiken om terug te keren naar Screen1. Gebruik in plaats daarvan het blok.close screen

Open een ander scherm met startwaarde

Opent een ander scherm en geeft er een waarde aan door.

Tekst voor het starten van een opmaak ophalen

Retourneert de tekst zonder opmaak die aan dit scherm is doorgegeven toen het door een andere app werd gestart. Als er geen waarde is doorgegeven, wordt de lege tekst geretourneerd. Gebruik voor apps met meerdere schermen in plaats van .get start valueget plain start tekst

Krijg startwaarde

Retourneert de startwaarde die aan het huidige scherm is gegeven.

Deze waarde wordt gegeven door het gebruik van of .open another screen with start valueclose screen with value

Scherm sluiten

Hiermee sluit u het huidige scherm.

Sluit het scherm met platte tekst

Sluit het huidige scherm en geeft tekst door aan de app die dit scherm heeft geopend. Deze opdracht is bedoeld voor het retourneren van tekst naar niet-App Inventor-activiteiten, niet naar App Inventor-schermen. Voor App Inventor-schermen, zoals in apps met meerdere schermen, gebruikt u , niet .close screen with valueclose screen with plain tekst

Sluit het scherm met waarde

Hiermee sluit u het huidige scherm en retourneert u een waarde naar het scherm waarmee dit scherm is geopend.

Applicatie sluiten

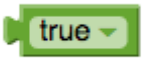
Sluit de toepassing.

breken

Bij het herhalen met behulp van het for-bereik, voor elk of while blokken is het soms handig om de lus vroegtijdig te kunnen verlaten. Hiermee kunt u aan de lus ontsnappen. Wanneer dit wordt uitgevoerd, verlaat dit de lus en gaat de app verder met de instructies die na de lus in de blokken voorkomen.break

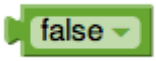
Logische Blokken

waar



Vertegenwoordigt de constante waarde waar. Gebruik het voor het instellen van booleaanse eigenschapswaarden van componenten, of als de waarde van een variabele die een voorwaarde vertegenwoordigt.

vals



Vertegenwoordigt de constante waarde false. Gebruik het voor het instellen van booleaanse eigenschapswaarden van componenten, of als de waarde van een variabele die een voorwaarde vertegenwoordigt.

niet

Voert logische ontkenning uit, waarbij false wordt geretourneerd als de invoer waar is en true als de invoer onwaar is.

= {#=}

Test of de argumenten gelijk zijn.

Twee getallen zijn gelijk als ze numeriek gelijk zijn, bijvoorbeeld 1 is gelijk aan 1,0.

Twee tekstblokken zijn gelijk als ze dezelfde tekens hebben in dezelfde volgorde, met hetzelfde hoofdlettergebruik. Banaan is bijvoorbeeld niet gelijk aan Banaan.

Getallen en tekst zijn gelijk als het getal numeriek gelijk is aan een getal dat bij die tekst zou worden afgedrukt. Bijvoorbeeld, 12,0 is gelijk aan het resultaat van het samenvoegen van het eerste teken van 1A met het laatste teken van Teafor2.

Twee lijsten zijn gelijk als ze hetzelfde aantal elementen hebben en de corresponderende elementen gelijk zijn.

Werkt precies hetzelfde als het =-blok in wiskunde

≠ {#not=}

Tests om te zien of twee argumenten niet gelijk zijn.

en

Hiermee wordt getest of alle logische voorwaarden waar zijn. Het resultaat is waar dan en slechts dan als alle geteste voorwaarden waar zijn. Het aantal tests kan worden uitgebreid met behulp van de mutator. De condities worden van links naar rechts getest en het testen stopt zodra een van de condities vals is. Als er geen voorwaarden zijn om te testen, dan is het resultaat waar. Je kunt dit beschouwen als een grap van een logicus.

of

Hiermee wordt getest of een van de logische voorwaarden waar is. Het resultaat is waar als een of meer van de geteste voorwaarden waar zijn. Het aantal tests kan worden uitgebreid met behulp van de mutator. De condities worden van links naar rechts getest en het testen stopt zodra een van de condities waar is. Als er geen voorwaarden zijn om te testen, is het resultaat vals.

=, ≠, >, ≥, <, ≤

min, max

Sqrt, ABS, -, log, E^, Rond, plafond, vloer

modulo van, rest van, quotiënt van

zonde, cos, tanen, asin, acos, atan

Converteer radialen naar graden, converteer graden naar radialen

Wiskundige Blokken

Basis Nummer Blok

Kan worden gebruikt als elk positief of negatief getal. Door op de "0" in het blok te klikken, kunt u het nummer wijzigen.

Het blok ondersteunt normale getallen met grondtal 10 (bijvoorbeeld: 2, 12 en 2.12), evenals C-achtige voorvoegsels voor andere getallenstelsels. Het ondersteunt:

Grondtal 2 (binaire) getallen, bijv. 0b10 (decimaal 2)

Grondtal 8 (octaale) getallen, bijv. 0o14 (decimaal 12)

Grondtal 16 (hexadecimale) getallen, bijv. 0xd4 (decimaal 212)

Radix Nummer Blok

Vertegenwoordigt een getal met grondtal 10. Door op de "0" te klikken, kunt u het nummer wijzigen.

Als u op de vervolgkeuzelijst klikt, kunt u een getal invoeren in een ander getallenstelsel (ook wel radix genoemd). Het getal wordt dan "vertaald" in decimaal (ook wel grondtal 10 genoemd).

Deze drie blokken zijn bijvoorbeeld gelijkwaardig:

De vervolgkeuzelijst ondersteunt: decimale (grondtal 10), binaire (grondtal 2), octale (grondtal 8) en hexadecimale (grondtal 16) invoerformaten.

Met de decimale modus kunt u elk positief of negatief getal invoeren (bijv. 2, -12, 2,12). In de andere modi kunt u alleen een geheel getal invoeren (ook wel een positief getal of nul genoemd).

= {#=}

Test of twee getallen gelijk zijn en retourneert waar of onwaar.

≠ {#not=}

Test of twee getallen niet gelijk zijn en retourneert waar of onwaar.

>

Hiermee wordt getest of het eerste getal groter is dan het tweede getal en waar of onwaar retourneert.

≥

Test of het eerste getal groter is dan of gelijk is aan het tweede getal en waar of onwaar retourneert.

<

Hiermee wordt getest of het eerste getal kleiner is dan het tweede getal en waar of onwaar retourneert.

≤

Hiermee wordt getest of het eerste getal kleiner is dan of gelijk is aan het tweede getal en waar of onwaar retourneert.

+

Retourneert het resultaat van het optellen van een willekeurig aantal blokken met een getalwaarde. Blokken met een getalwaarde omvatten het basisgetallenblok, de lengte van de lijst of tekst, variabelen met een getalwaarde, enz. Dit blok is een mutator en kan worden uitgebreid om meer getallen in de som toe te staan.

-

*

Retourneert het resultaat van het vermenigvuldigen van een willekeurig aantal blokken met een getalwaarde bij elkaar. Het is een mutatorblok en kan worden uitgebreid om meer nummers in het product toe te staan.

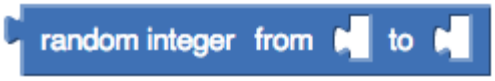
/

Berekent het resultaat van het delen van het eerste getal door het tweede.

^

Retourneert het resultaat van het eerste getal dat tot de macht van het tweede getal wordt verhoogd.

Willekeurig geheel getal



Retourneert een willekeurig geheel getal tussen de opgegeven waarden, inclusief. De volgorde van de argumenten doet er niet toe.

willekeurige breuk

Retourneert een willekeurige waarde tussen 0 en 1.

willekeurige set zaad naar

Gebruik dit blok om herhaalbare reeksen van willekeurige getallen te genereren. U kunt dezelfde reeks willekeurige getallen genereren door eerst een willekeurige set seed met dezelfde waarde aan te roepen. Dit is handig voor het testen van programma's met willekeurige waarden.

Min

Retourneert de kleinste waarde van een reeks getallen. Als er niet-aangesloten stopcontacten in het blok zijn, zal min ook rekening houden met 0 in de reeks getallen. Dit blok is een mutator en een dropdown.

Max

Retourneert de grootste waarde van een reeks getallen. Als er niet-aangesloten stopcontacten in het blok zijn, zal max ook rekening houden met 0 in zijn reeks getallen. Dit blok is een mutator en een dropdown.

vierkantswortel

Berekent de vierkantswortel van het opgegeven getal.

ABS

Berekent de absolute waarde van het opgegeven getal.

Neg

Berekent het negatieve getal van een bepaald getal.

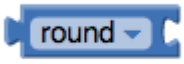
log

Berekent de natuurlijke logaritme van een gegeven getal, dat wil zeggen de logaritme naar het grondtal e (2,71828...).

e^

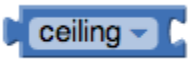
Retourneert e (2,71828...) verheven tot de macht van het gegeven getal.

rond



Retourneert het opgegeven getal afgerond op het dichtstbijzijnde gehele getal. Als het breukdeel < .5 is, wordt het naar beneden afgerond. Als het > .5 is, wordt het naar boven afgerond. Als het precies gelijk is aan .5, worden getallen met een even geheel deel naar beneden afgerond en getallen met een oneven geheel deel naar boven afgerond. (Deze methode wordt rond tot even genoemd.)

plafond

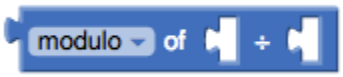


Retourneert het kleinste gehele getal dat groter is dan of gelijk is aan het opgegeven getal.

vloer

Retourneert het grootste gehele getal dat kleiner is dan of gelijk is aan het opgegeven getal.

formulier



Modulo(a,b) is hetzelfde als rest(a,b) wanneer a en b positief zijn. Meer in het algemeen wordt modulo(a,b) gedefinieerd voor elke a en b, zodat (floor(a/b)× b) + modulo(a,b) = a. Bijvoorbeeld: modulo(11, 5) = 1, modulo(-11, 5) = 4, modulo(11, -5) = -4, modulo(-11, -5) = -1. Modulo(a,b) heeft altijd hetzelfde teken als b, terwijl rest(a,b) altijd hetzelfde teken heeft als a.

rest

Rest(a,b) retourneert het resultaat van het delen van a door b en het nemen van de rest. De rest is het fractionele deel van het resultaat vermenigvuldigd met b.

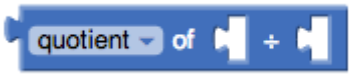
Bijvoorbeeld, rest(11,5) = 1 omdat

11 / 5 = 2 1⁄5

In dit geval, 1⁄5 is het fractionele deel. We vermenigvuldigen dit met b, in dit geval 5 en we krijgen 1, onze rest.

Andere voorbeelden zijn rest(-11, 5) = -1, rest(11, -5) = 1 en rest(-11, -5) = -1.

quotiënt



Berekent het resultaat van het delen van het eerste getal door het tweede getal en het weggooien van een breukdeel van het resultaat.

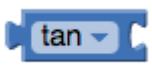
zonder

Berekent de sinus van het opgegeven getal in graden.

COS

Berekent de cosinus van het opgegeven getal in graden.

raaklijn



Berekent de raaklijn van het opgegeven getal in graden.

Boogsinus

Berekent de boogsinus van het opgegeven getal in graden.

Boogcosinus

Berekent de boogcosinus van het opgegeven getal in graden.

Atan

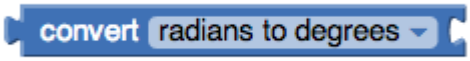
Boogtangens

Berekent de boogtangens van het opgegeven getal in graden.

Boogtangens

Berekent de boogtangens van y/x, gegeven y en x.

Radialen omzetten in graden

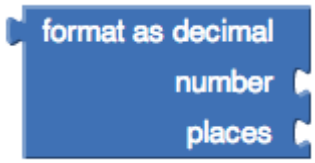


Berekent de waarde in graden van het opgegeven getal in radialen. Het resultaat is een hoek in het bereik [0, 360]

Graden converteren naar radialen

Berekent de waarde in radialen van het opgegeven getal in graden. Het resultaat is een hoek in het bereik [-π , +π)

Opmaken als decimaal

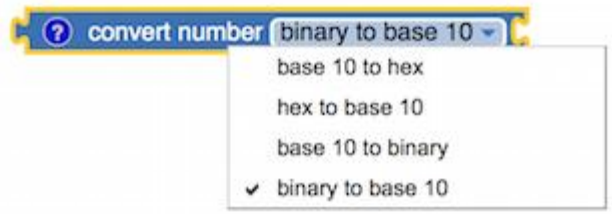


Hiermee maakt u een getal op als een decimaal met een bepaald aantal plaatsen achter de komma. Het aantal plaatsen moet een niet-negatief geheel getal zijn. Het resultaat wordt verkregen door het getal af te ronden (als er te veel plaatsen waren) of door nullen aan de rechterkant toe te voegen (als er te weinig waren).

is een getal?

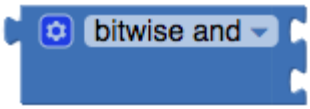
Retourneert true als het opgegeven object een getal is, en false als dat niet het geval is.

Nummer converteren



Neemt een tekenreeks die een positief geheel getal in één basis vertegenwoordigt en retourneert een tekenreeks die hetzelfde getal vertegenwoordigt in een ander grondtal. Als de invoertekenreeks bijvoorbeeld 10 is, zal het converteren van grondtal 10 naar binair de tekenreeks 1010 opleveren; terwijl als de invoertekenreeks dezelfde 10 is, het converteren van binair naar grondtal 10 de tekenreeks 2 zal produceren. Als de invoertekenreeks dezelfde 10 is, zal het converteren van grondtal 10 naar zeskant de tekenreeks A opleveren.

Bitwise en

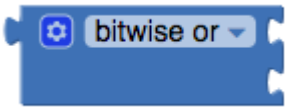


Neemt twee getallen en vergelijkt elk paar bits. Elk bit van het resultaat is alleen 1 als de corresponderende bits van beide operanden 1 zijn.

Voorbeeld:

	Decimaal	Binair (interne representatie)
	6	0 1 1 0
	3	0 0 1 1
Resultaat:	2	0 0 1 0

Bitsgewijs of (inclusief)



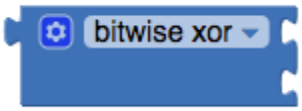
Neemt twee getallen en vergelijkt elk paar bits. Elk bit van het resultaat is 1 als een van de corresponderende bits in elke operand 1 is.

Voorbeeld:

	Decimaal	Binair (interne representatie)
	6	0 1 1 0
	3	0 0 1 1
Resultaat:	7	0 1 1 1

Bitwise

Or (Exclusief)



Neemt twee getallen en vergelijkt elk paar bits. Elk bit van het resultaat is alleen 1 als een corresponderend bit in de operanden 1 is en het andere 0.

Voorbeeld:

	Decimaal	Binair (interne representatie)
	6	0 1 1 0
	3	0 0 1 1
Resultaat:	5	0 1 0 1

Berekent het rekenkundig gemiddelde van de eleme.nten in een lijst.

Retourneert het minimumelement in een lijst.

Retourneert het maximale element in een lijst.

Retourneert het meest voorkomende element in een lijst.

Berekent het geometrisch gemiddelde van de elementen in een lijst.

Berekent de (populatie) standaarddeviaties van de elementen in een lijst.

Berekent de standaardfout van de elementen in een lijst.

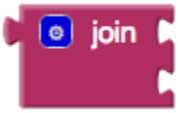
Tekstblokken

" " (snaarblok)

Bevat een tekenreeks.

Deze tekenreeks kan alle tekens bevatten (letters, cijfers of andere speciale tekens). In App Inventor wordt het beschouwd als een tekstobject.

verbinden



Voegt alle ingangen toe om een enkele tekenreeks te maken. Als er geen invoer is, wordt een lege tekenreeks geretourneerd.

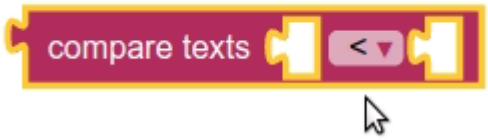
lengte

Retourneert het aantal tekens inclusief spaties in de tekenreeks. Dit is de lengte van de gegeven tekstreeks.

is leeg

Geeft aan of de tekenreeks al dan niet tekens bevat (inclusief spaties). Als de lengte van de tekenreeks 0 is, retourneert true anders onwaar.

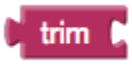
Vergelijk teksten < > = ≠



Geeft als resultaat of de eerste tekenreeks lexicografisch <, >, = of ≠ de tweede tekenreeks is, afhankelijk van de vervolgkeuzelijst die is geselecteerd.

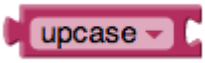
Een tekenreeks wordt lexicografisch als groter beschouwd dan een andere als deze alfabetisch groter is dan de andere tekenreeks. In wezen zou het erna komen in het woordenboek. Alle hoofdletters worden als kleiner beschouwd of komen voor kleine letters voor. kat zou > kat zijn.

bijknippen



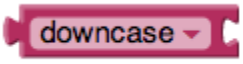
Hiermee verwijdert u alle spaties die aan of achter de invoertekenreeks staan en retourneert u het resultaat.

Bovenkast



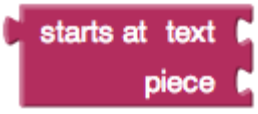
Retourneert een kopie van het argument van de tekenreeks die is geconverteerd naar hoofdletters.

Neerwaartse bijstelling



Returns a copy of its text string argument converted to all lower case.

begint bij



Retourneert de tekenpositie waar het eerste teken van het stuk voor het eerst in de tekst voorkomt, of 0 als deze niet aanwezig is. Bijvoorbeeld, de locatie van ana in havana banaan is 4.

Bevat

Retourneert true als het stuk in de tekst voorkomt; anders retourneert false.

bevat alle

Retourneert true als een van de stukken in de stuklijst in tekst voorkomt; anders retourneert false.

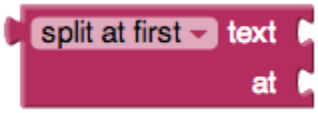
Dit blok kan worden verkregen door de vervolgkeuzelijst op het bevat-blok te wijzigen.

Bevat alle

Retourneert true als alle stukken in de stuklijst in tekst voorkomen; anders retourneert false.

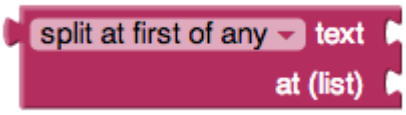
Dit blok kan worden verkregen door de vervolgkeuzelijst op het bevat-blok te wijzigen.

In eerste instantie gesplitst



Verdeelt de gegeven tekst in twee stukken met behulp van de locatie van het eerste voorkomen van at als scheidingspunt, en retourneert een lijst met twee items bestaande uit het stuk vóór het scheidingspunt en het stuk na het scheidingspunt. Het splitsen van appel, banaan, kers, hondenvoer met een komma als splitspunt retourneert een lijst met twee items: de eerste is de tekst appel en de tweede is de tekst banaan, kers, hondenvoer. Merk op dat de komma na appel niet in het resultaat voorkomt, want dat is het scheidingspunt.

splitsen bij de eerste van elke



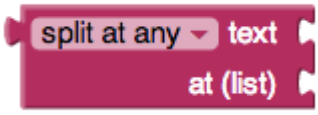
Verdeelt de gegeven tekst in een lijst met twee items, waarbij de eerste locatie van een item in de lijst wordt gebruikt als scheidingspunt.

Het splitsen van i love apples bananas appels druiven door de lijst [ba,ap] zou resulteren in een lijst van twee items, de eerste is i love en de tweede ples bananen appels druiven.

splijten

Verdeelt tekst in stukken met behulp van at als scheidingspunten en produceert een lijst met de resultaten. Splitsen van een, twee, drie, vier op , (komma) retourneert de lijst (één twee drie vier). Als je één-aardappel, twee-aardappel, drie-aardappel, vier op -aardappel splitst, retourneert de lijst (één twee drie vier).

splitsen op elk moment



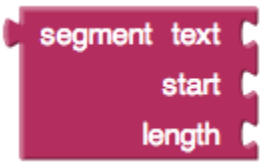
Verdeelt de gegeven tekst in een lijst, waarbij een van de items in at als scheidingspunt wordt gebruikt, en retourneert een lijst met de resultaten.

Het splitsen van appelbes, banaan, kers, hondenvoer met at als de lijst met twee elementen waarvan het eerste item een komma is en het tweede item rry is, retourneert een lijst met vier items: (applebe banana che dogfood).

splitsen bij spaties

Verdeelt de gegeven tekst bij elk voorkomen van een spatie en produceert een lijst van de stukken.

segment



Extraheert een deel van de tekst vanaf de beginpositie en gaat door voor lange tekens.

Vervang alles

Retourneert een nieuwe tekenreeks die is verkregen door alle exemplaren van de subtekenreeks te vervangen door de vervanging.

Vervang alles door Ze houdt van eten. Ze houdt van schrijven. Ze houdt van coderen als de tekst, zij als het segment, en Hannah als de vervanger zou ertoe leiden dat Hannah van eten houdt. Hannah houdt van schrijven. Hannah houdt van coderen.

Notitie: Reguliere expressies (regex) zijn patronen die worden gebruikt voor het matchen van tekencombinaties in tekenreeksen. Bij het gebruik van het blok "alles vervangen" is het essentieel om op de hoogte te zijn van speciale tekens in reguliere expressies. Speciale tekens zoals $, *, ., enz. hebben een speciale betekenis in reguliere expressies. Als je deze tekens als letterlijke tekens in je patroon wilt gebruiken, moet je ze escapen met een backslash (\).

Voorbeeld: vervang alles door Prijs: $ 50,00 als tekst, $ als segment, € als vervanging dit zou resulteren in Prijs: € 50,00

Versluierde tekst

Produceert tekst, zoals een tekstblok. Het verschil is dat de tekst niet gemakkelijk te vinden is door de inhoud van de app te onderzoeken. Gebruik dit bij het maken van apps om te distribueren die vertrouwelijke informatie bevatten, bijvoorbeeld API-sleutels.

Waarschuwing: Dit biedt slechts een zeer lage beveiliging tegen deskundige tegenstanders.

is een snaar?

Retourneert waar als het ding een tekstobject is, anders onwaar.

omkeren

Draai de gegeven tekst om. "Reverse" zou bijvoorbeeld "esrever" worden.

Vervang alle toewijzingen

Met een woordenboek met toewijzingen als invoer, worden de sleutelitems in de tekst vervangen door de corresponderende waarden in de woordenlijst. Retourneert de tekst waarop de toewijzingen zijn toegepast.

Woordenboek volgorde

Als de volgorde van de woordenlijst is opgegeven, in het geval dat een sleutelvermelding een subtekenreeks is van een andere sleutelvermelding, moet de eerste worden vervangen is gebaseerd op de invoervolgorde in het woordenboek (de vroegste wordt als eerste vervangen).

Langste snaar eerste orde

Als de langste tekenreeksvolgorde is opgegeven, in het geval dat een sleutelinvoer een subtekenreeks is van een andere sleutelinvoer, de eerste die moet worden vervangen is degene die langer is.

Lijst Blokken

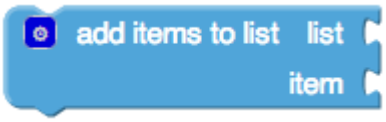
Lege lijst maken

Hiermee maakt u een lege lijst zonder elementen.

Maak een lijst

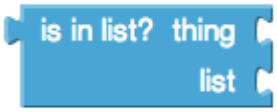
Maakt een lijst van de opgegeven blokken. Als u geen argumenten opgeeft, ontstaat er een lege lijst, waaraan u later elementen kunt toevoegen. Dit blok is een mutator. Als u op het blauwe plusteken klikt, kunt u extra items aan uw lijst toevoegen.

Items toevoegen aan lijst



Voegt de gegeven items toe aan het einde van de lijst. Het verschil tussen dit en toevoegen aan lijst is dat toevoegen aan lijst de items die moeten worden toegevoegd als één lijst neemt terwijl items toevoegen aan lijst de items als afzonderlijke argumenten gebruikt. Dit blok is een mutator.

staat in de lijst



Als ding een van de elementen van de lijst is, retourneert true; anders retourneert false. Merk op dat als een lijst sublijsten bevat, De leden van de sublijsten zijn zelf geen lid van de lijst. De leden van de lijst (1 2 (3 4)) zijn bijvoorbeeld 1, 2 en de lijst (3 4); 3 en 4 zijn zelf geen lid van de lijst.

Lengte van de lijst

Retourneert het aantal items in de lijst.

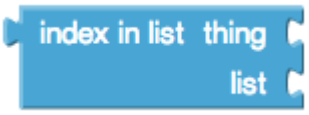
is de lijst leeg

Als de lijst geen items bevat, wordt waar geretourneerd. anders retourneert false.

Kies een willekeurig item

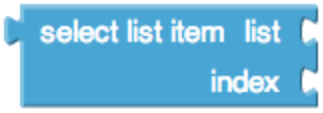
Kiest willekeurig een item uit de lijst.

index in lijst



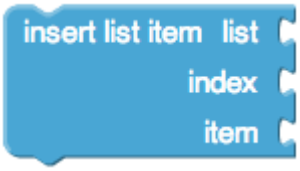
Berekent de positie van het ding in de lijst. Als deze niet in de lijst staat, wordt 0 geretourneerd.

Lijstitem selecteren



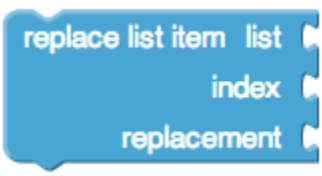
Selecteert het item in de opgegeven index in de gegeven lijst. Het eerste item in de lijst bevindt zich in index 1.

Lijstitem invoegen



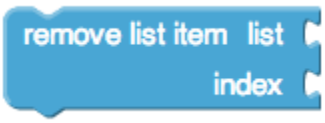
Hiermee voegt u een item in de lijst op de opgegeven positie in.

Lijstitem vervangen



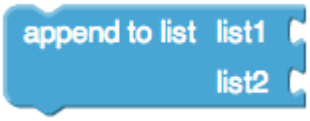
Voegt vervanging toe aan de gegeven lijst bij positie-index. Het vorige item op die positie wordt verwijderd.

Lijstitem verwijderen



Verwijdert het item op de opgegeven positie.

Toevoegen aan lijst



Hiermee voegt u de items in de tweede lijst toe aan het einde van de eerste lijst.

Lijst kopiëren

Maakt een kopie van een lijst, inclusief het kopiëren van alle sublijsten.

is een lijst?

Als het een lijst is, retourneert het waar; anders retourneert false.

Omgekeerdelijst

Retourneert een kopie van de lijst met items in omgekeerde volgorde. Bijvoorbeeld: reverse([1,2,3]) retourneert [3,2,1]

Lijst naar csv-rij

Interpreteert de lijst als een rij van een tabel en retourneert een CSV-tekst (door komma's gescheiden waarden) die de rij vertegenwoordigt. Elk item in de rijenlijst wordt beschouwd als een veld en wordt tussen dubbele aanhalingstekens geplaatst in de resulterende CSV-tekst. Items worden gescheiden door komma's. Als u bijvoorbeeld de lijst (a, b, c, d) converteert naar een CSV-rij, krijgt u ('a', 'b', 'c', 'd'). De geretourneerde rijtekst heeft geen regelscheidingsteken aan het einde.

Lijst naar CSV-tabel

Interpreteert de lijst als een tabel in de indeling van de rij en retourneert een CSV-tekst (door komma's gescheiden waarden) die de tabel vertegenwoordigt. Elk item in de lijst moet zelf een lijst zijn die een rij van de CSV-tabel vertegenwoordigt. Elk item in de rijenlijst wordt beschouwd als een veld en wordt tussen dubbele aanhalingstekens geplaatst in de resulterende CSV-tekst. In de geretourneerde tekst worden items in rijen gescheiden door komma's en rijen gescheiden door CRLF (\r\n).

Lijst uit csv-rij

Parseert een tekst als een door komma's gescheiden rij (CSV-opgemaakte waarden) om een lijst met velden te produceren. Als u bijvoorbeeld ('a', 'b', 'c', 'd') converteert naar een lijst, krijgt u (a b c d).

Lijst uit CSV-tabel

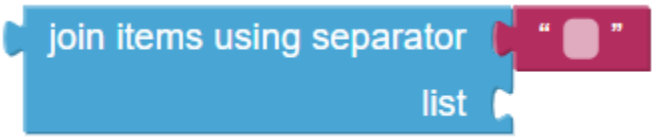
Parseert een tekst als een CSV-tabel (door komma's gescheiden waarden) om een lijst met rijen te produceren, die elk een lijst met velden zijn. Rijen kunnen worden gescheiden door nieuwe regels (\n) of CRLF (\r\n).

Opzoeken in paren

Wordt gebruikt voor het opzoeken van informatie in een woordenboekachtige structuur die wordt weergegeven als een lijst. Voor deze bewerking zijn drie invoergegevens nodig, een sleutel, een lijstpaar en een notFound-resultaat, dat standaard is ingesteld op 'not found'. Hier moeten paren een lijst van paren zijn, dat wil zeggen een lijst waarbij elk element zelf een lijst van twee elementen is. Vindt het eerste paar in de lijst waarvan het eerste element de sleutel is, en retourneert het tweede element. Als de lijst bijvoorbeeld ((een appel) (d draak) (b goederenwagon) (cat 100)) is, dan levert het opzoeken van 'b' 'goederenwagon' op. Als er geen dergelijk paar in de lijst staat, retourneert het de parameter notFound.

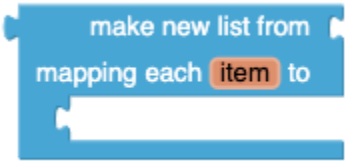
Als paren geen lijst is van koppelt, dan signaleert de bewerking een fout.Lookup in pairslookup in pairs

Samenvoegen met scheidingsteken



Voegt alle elementen in de opgegeven lijst samen met het opgegeven scheidingsteken, waardoor tekst wordt geproduceerd.

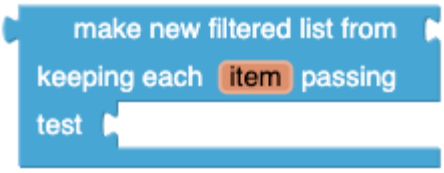
Maak een nieuwe lijst met toegewezen kaarten



Hiermee maakt u een nieuwe lijst door elk item in de invoerlijst toe te wijzen aan een nieuwe waarde met behulp van de opgegeven expressie. Het lichaam is een uitdrukking die elk item in de lijst manipuleert. Gebruik de opgegeven variabelenaam, item, om naar het huidige lijstitem te verwijzen.

Hier is een tutorial over het gebruik van het kaartblok.

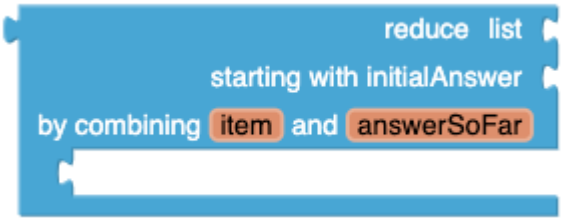
Nieuwe gefilterde lijst maken



Maak een nieuwe lijst door elk item in de invoerlijst te laten voldoen aan de test. Het lichaam is een booleaanse uitdrukking die controleert of een item de test doorstaat. Als de hoofdtekst true retourneert, wordt het item toegevoegd aan de nieuwe gefilterde lijst. Gebruik de opgegeven variabelenaam, item, om naar het huidige lijstitem te verwijzen.

Hier is een tutorial over het gebruik van het filterblok.

Reduceer de lijst tot één waarde



Retourneert een geaccumuleerde waarde door de invoerlijst te verkleinen. Als de invoerlijst leeg is, wordt initialAnswer geretourneerd. Anders wordt answerSoFar geïnitialiseerd naar initialAnswer. Het hoofdblok wordt geëvalueerd met behulp van het geaccumuleerde antwoordSoFar en elk item in de invoerlijst.

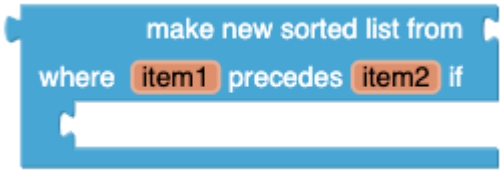
Hier is een tutorial over het gebruik van het reduceerblok.

Lijst sorteren in oplopende volgorde

Maak een nieuwe lijst door de invoerlijst in oplopende volgorde te sorteren. Dit is een generieke sorteerprocedure die werkt op lijsten van elk type. Het groepeert items van hetzelfde type en sorteert vervolgens dienovereenkomstig binnen dezelfde typegroep. De huidige volgorde van de typen is booleans, getallen, strings, lijsten en vervolgens componenten. Voor booleans wordt false gedefinieerd als minder dan true. Componenten worden eerst vergeleken met hun klassenamen. Als het exemplaren van dezelfde klasse zijn, worden hun hashcodes gebruikt om te vergelijken.

Hier is een tutorial over het gebruik van het sorteerblok.

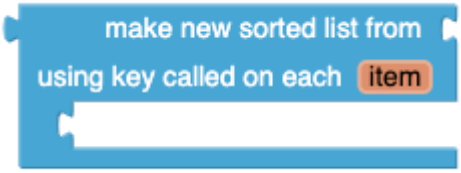
Lijst sorteren met een opgegeven vergelijker



Maak een nieuwe lijst door de invoerlijst te sorteren in een volgorde die is opgegeven door de hoofdtekst van het blok. De hoofdtekst van dit blok is een booleaanse expressie met item1 en item2 en retourneert waar of onwaar. Als de hoofdtekst waar retourneert, wordt item1 samengevoegd vóór item2 bij het sorteren. Als de hoofdtekst false retourneert, wordt item2 samengevoegd vóór item1 bij het sorteren. Gebruik de opgegeven variabelenamen, item1 en item2, om te verwijzen naar de twee huidige lijstitems die worden vergeleken.

Hier is een tutorial over het gebruik van het sorteerblok met vergelijkingsblok.

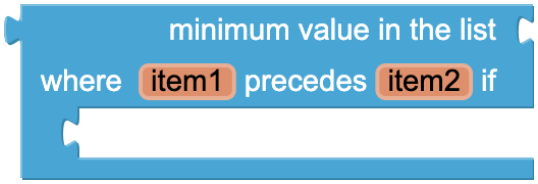
Lijst sorteren met sleutel



Maak een nieuwe lijst door de invoerlijst te sorteren met de toetsen in oplopende volgorde. De sleutels zijn proxywaarden die door de hoofdtekst van dit blok worden gegenereerd op basis van elk item in de lijst.

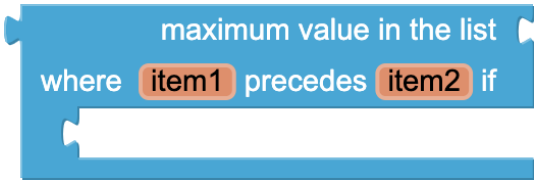
Hier is een tutorial over het gebruik van het sorteren met sleutelblok.

Minimumwaarde in lijst



Retourneert het minimumaantal in de invoerlijst.

Maximale waarde in lijst



Retourneert het maximale aantal in de invoerlijst.

Alles behalve eerst

Retourneer een lijst zonder het eerste item in de invoerlijst.

alles behalve de laatste

Retourneert een lijst zonder het laatste item in de invoerlijst.

Plakjes opsommen

Retourneer een lijst door de invoerlijst te segmenteren bij de twee opgegeven indexen. De geretourneerde lijst bevat items van de invoerlijst die begint bij index1 tot en met, maar exclusief index2.

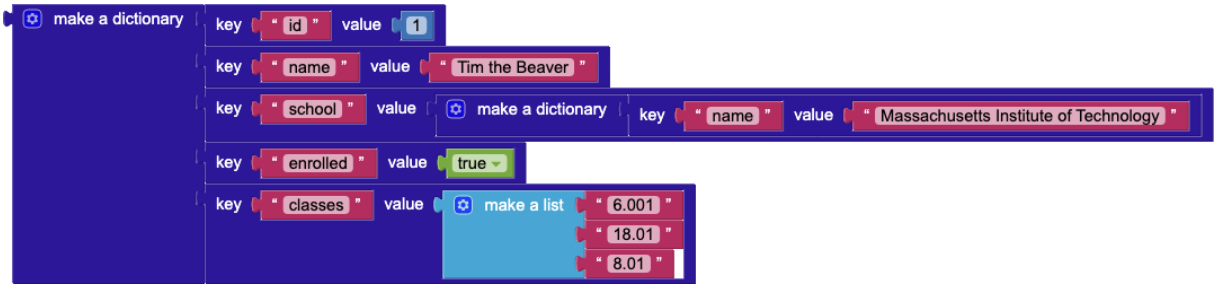
Woordenboek Blokken

Introductie

Woordenboeken, in andere talen termen genoemd zoals kaarten, associatieve matrices of lijsten, zijn gegevensstructuren die de ene waarde, vaak de sleutel genoemd, associëren met een andere waarde. Een veelgebruikte manier om woordenboeken weer te geven is het gebruik van de JavaScript Object Notation (JSON), bijvoorbeeld:

```
{
 "id":  1,
 "name":  "Tim the Beaver",
 "school": {
  "name": "Massachusetts Institute of Technology"
 },
 "enrolled": true,
 "classes": ["6.001", "18.01", "8.01"]
}
```

Het bovenstaande voorbeeld laat zien dat in JSON de sleutels (tussen aanhalingstekens voor de) kunnen worden toegewezen aan verschillende soorten waarden. De toegestane typen zijn getal, tekst, andere woordenboeken, booleans en lijsten. In de blokkentaal kun je dit woordenboek als volgt gebruiken::



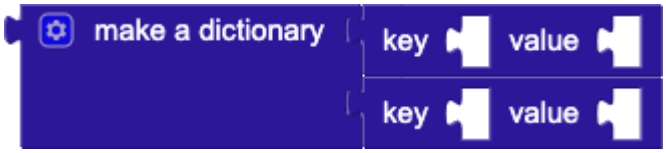
Afbeelding 1: Een blokweergave van het hierboven getoonde JSON-codefragment.

Leeg woordenboek maken

Het blok maakt een woordenboek zonder sleutel-waardeparen. Items kunnen met behulp van het blok aan het lege woordenboek worden toegevoegd. Het blok kan ook in een blok worden

veranderd door de blauwe mutatorknop te gebruiken om items toe te voegen.create empty dictionaryset value for keycreate empty dictionarymake a dictionarypair

Maak een woordenboek



Het wordt gebruikt om een woordenboek te maken met een set s die van tevoren bekend is. Extra items kunnen worden toegevoegd met behulp van .make a dictionarypairset value for key

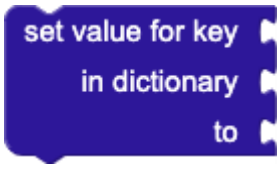
paar

Het blok is een blok voor speciale doeleinden dat wordt gebruikt voor het samenstellen van woordenboeken.pair

Krijg waar voor je geld

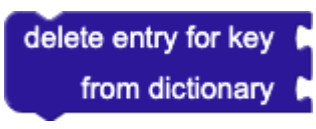
Het blok controleert of het woordenboek een overeenkomstige waarde bevat voor de gegeven sleutel. Als dit het geval is, wordt de waarde geretourneerd. Anders wordt de waarde van de parameter geretourneerd. Dit gedrag is vergelijkbaar met het gedrag van het blok Opzoeken in paren.get value for keynot found

Waarde instellen voor sleutel



Het blok stelt de corresponderende waarde in voor de gegeven in de to . Als er geen toewijzing bestaat voor , wordt er een nieuwe gemaakt. Anders wordt de bestaande waarde vervangen door de nieuwe waarde.set value for keykeydictionaryvaluekey

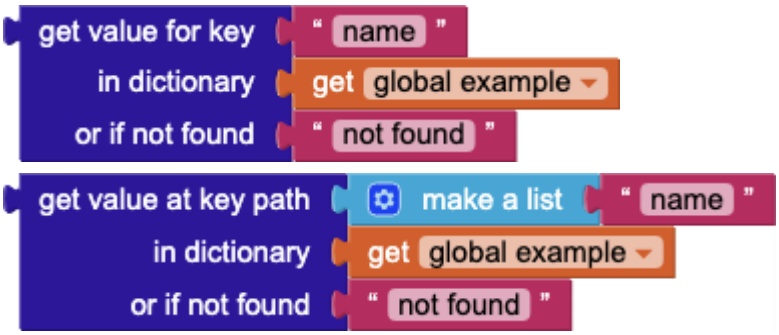
Invoer voor sleutel verwijderen



Het blok verwijdert de sleutel-waardetoewijzing in de woordenlijst voor de opgegeven sleutel. Als er geen vermelding voor de sleutel in het woordenboek bestaat, wordt het woordenboek niet gewijzigd.delete entry for key

Krijg waarde op het belangrijkste pad

Het blok is een meer geavanceerde versie van het blok. In plaats van de waarde van een specifieke sleutel te krijgen, wordt in plaats daarvan een lijst met geldige sleutels en nummers gebruikt die een pad door een gegevensstructuur vertegenwoordigen. Het blok is gelijk aan het gebruik van dit blok met een sleutelpad van lengte 1 dat de sleutel bevat. De volgende twee blokken zouden bijvoorbeeld terugkeren:get value at key pathget value for keyget value for key"Tim the Beaver"



Het doorloopt de gegevensstructuur, beginnend bij de oorspronkelijke woordenlijst, met behulp van de verstrekte om waarden op te halen die diep in complexe gegevensstructuren zijn genest. Het wordt het best gebruikt voor het verwerken van JSON-gegevens van webservices. Uitgaande van de eerste invoer, neemt het het eerste element in de en controleert of er op dat moment een sleutel (als de invoer een woordenboek is) of index (als de invoer een lijst is) bestaat. Als dit het geval is, selecteert het dat item als invoer en gaat het verder met het controleren van het volgende element in de , totdat ofwel het hele pad is gevolgd, waarna het retourneert wat zich op die locatie bevindt, of de parameter.pathkey pathkey path"not found"

Voorbeelden

{

"id": 1,

"name": "Tim the Beaver",

"school": {

"name": "Massachusetts Institute of Technology"

},

"enrolled": true,

"classes": ["6.001", "18.01", "8.01"]

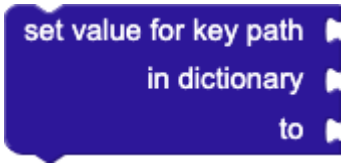
}

Bijvoorbeeld, gegeven het JSON-woordenboek hierboven, zal het volgende gebruik van het resultaat opleveren .get value at key path"Massachusetts Institute of Technology"

Hiermee kan het pad getallen bevatten die de index van elementen vertegenwoordigen die moeten worden doorlopen wanneer woordenboeken en lijsten worden gemengd. Als we bijvoorbeeld wilden weten welke tweede les Tim volgde, konden we het volgende doen:get value at key path‹

die de waarde retourneert ."18.01"

Waarde instellen voor sleutelpad



Het blok werkt de waarde bij op een specifiek in een gegevensstructuur. Het is de spiegel van , die een waarde ophaalt op een specifieke . Het pad moet geldig zijn, met uitzondering van de laatste sleutel, die, als er geen toewijzing bestaat, een toewijzing aan de nieuwe waarde zal maken. Anders wordt de bestaande waarde vervangen door de nieuwe waarde.set value for key pathkey pathget value for key pathkey path

Sleutels ophalen

Het retourneert een lijst met sleutels in het woordenboek.get keys

Waarden ophalen

Het retourneert een lijst met de waarden in het woordenboek. Als u de inhoud van een waarde in de lijst wijzigt, wordt deze ook in het woordenboek gewijzigd.get values

Is de sleutel in het woordenboek?

De test of de sleutel bestaat in het woordenboek en retourneert als dat het geval is, anders retourneert het .is key in dictionary?truefalse

Grootte van het woordenboek

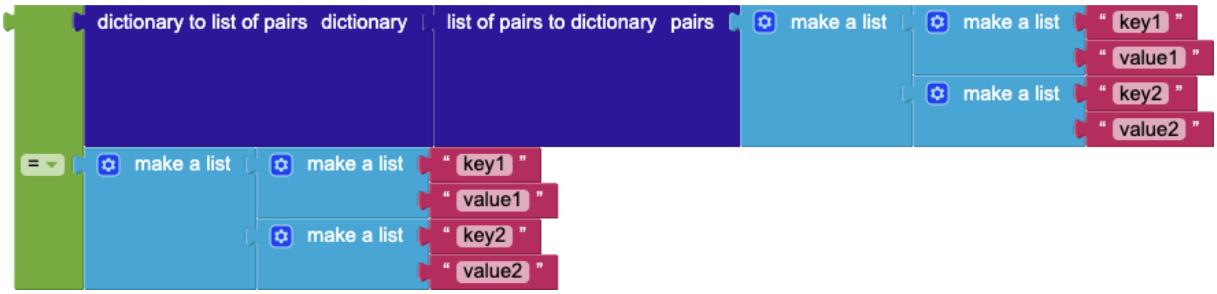
Het blok retourneert het aantal sleutel-waardeparen dat in de woordenlijst aanwezig is.size of dictionary

Lijst van paren naar woordenboek

Het blok converteert een associatieve lijst van het formulier in een woordenboek dat de sleutels aan hun waarden koppelt. Omdat woordenboeken betere opzoekprestaties bieden dan associatieve lijsten, is het raadzaam om dit blok te gebruiken om de associatieve lijst eerst om te zetten in een woordenboek als u veel bewerkingen op een gegevensstructuur wilt uitvoeren.list of pairs to dictionary((key1 value1) (key2 value2) ...)

woordenboek naar lijst van paren

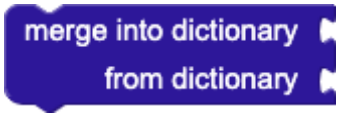
Het converteert een woordenboek in een associatieve lijst. Dit blok keert de conversie terug die wordt uitgevoerd door de lijst met paren naar het woordenboekblok.dictionary to list of pairs



Kopieer woordenboek

Het maakt een diepe kopie van het gegeven woordenboek. Dit betekent dat alle waarden recursief worden gekopieerd en dat het wijzigen van een waarde in de kopie deze niet wijzigt in het origineel.copy dictionary

Samenvoegen in woordenboek



Het blok kopieert de sleutel-waardeparen van de ene woordenlijst naar de andere, waarbij alle sleutels in de doelwoordenlijst worden overschreven.merge into dictionary from dictionary

Lijst op wandelsleutelpad

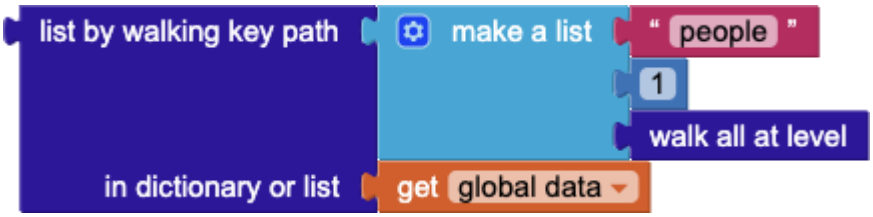
Het blok werkt op dezelfde manier als de , maar maakt een lijst met waarden in plaats van een enkele waarde te retourneren. Het werkt door te beginnen bij het gegeven woordenboek en langs de boom van objecten langs het gegeven pad naar beneden te lopen. In tegenstelling tot de though, kan het pad bestaan uit drie hoofdtypen: woordenboeksleutels, lijstindexen en de wandeling allemaal op niveaublok. Als er een sleutel of index wordt opgegeven, wordt het specifieke pad op dat punt in de boom genomen. Als de is opgegeven, wordt elke waarde op dat punt achter elkaar gevolgd (breedte-eerst), waarna de wandeling verder gaat vanaf het volgende element in het pad. Elk element dat overeenkomt met het hele pad wordt toegevoegd aan de uitvoerlijst.list by walking key pathget value at key pathget value at key pathwalk all at level

Voorbeelden

Denk aan de volgende JSON en blokken:

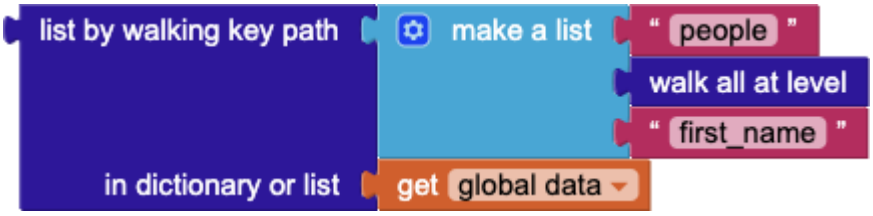
```
{
 "people": [{
  "first_name": "Tim",
  "last_name": "Beaver"
 },{
  "first_name": "John",
  "last_name": "Smith",
 },{
  "first_name": "Jane",
  "last_name": "Doe"
 }]
```

}



Als het een woordenboek bevat dat wordt vertegenwoordigd door de JSON, dan zal het blok de lijst produceren. Eerst wordt de waarde van de tag, dat wil zeggen de lijst met personen, gekozen. Vervolgens wordt het eerste element in de lijst gekozen. Ten slotte selecteert het blok Alles op niveau de waarden in het object op dat punt, dat wil zeggen de waarden en .global datalist by walking key path["Tim", "Beaver"]"people""Tim""Beaver"

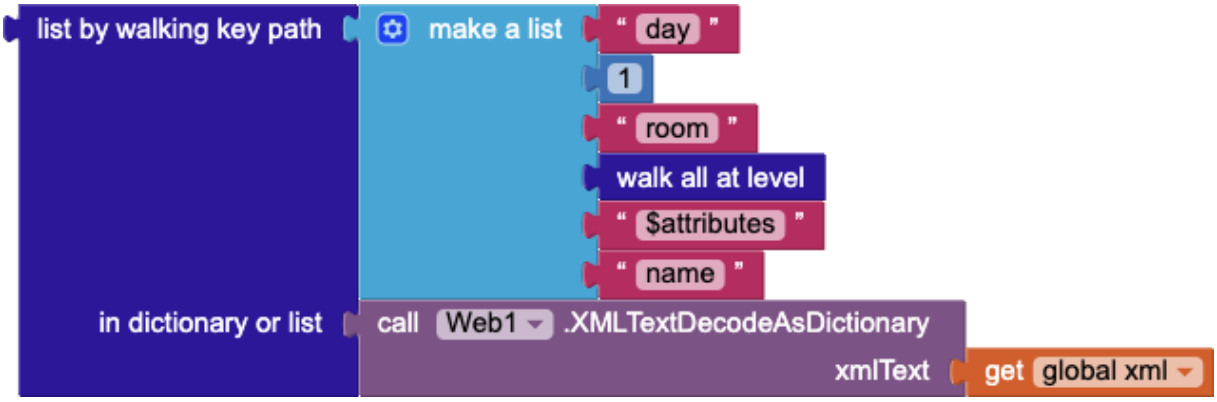
U kunt ook gebruiken op een niveau dat een lijst bevat. Het volgende blok selecteert bijvoorbeeld de voornamen van alle mensen in de structuur, d.w.z. .walk all at level["Tim", "John", "Jane"]



Dit blok kan ook worden gebruikt met XML die is geparseerd met behulp van het blok Web.XMLTextDecodeAsDictionary. Beschouw het volgende XML-document:

```
<schedule>
 <day>
  <room name="Hewlitt" />
  <room name="Bleil" />
 </day>
 <day>
  <room name="Kiva" />
  <room name="Star" />
 </day>
</schedule>
```

U kunt de volgende blokken gebruiken om een lijst te krijgen met de namen van de kamers op de eerste dag, d.w.z. .["Hewlitt", "Bleil"]



Loop allemaal op niveau

walk all at level Het blok is een gespecialiseerd blok dat kan worden gebruikt in het sleutelpad van een . Wanneer het tijdens een wandeling wordt aangetroffen, zorgt het ervoor dat elk item op dat niveau wordt verkend. Voor woordenboeken betekent dit dat elke waarde wordt bezocht. Voor lijsten wordt elk item in de lijst bezocht. Dit kan worden gebruikt om informatie samen te voegen uit een lijst met items in een woordenlijst, zoals de voornaam van elke persoon in een database die wordt vertegenwoordigd door JSON-objecten. Zie de lijst met het lopen van een sleutelpadblok voor voorbeelden.walk all at levellist by walking key path

is een woordenboek?

is a dictionary? Het blok test om te zien of het gegeven een woordenboek is of niet. Het zal terugkeren als het een woordenboek is en anderszins.is a dictionary?thingtruethingfalse

Kleurblokken

Er zijn drie hoofdtypen kleurblokken:

- een kleurenvak
- Kleur maken
- Gespleten kleur

Hoe werken kleuren in App Inventor?

Intern slaat App Inventor elke kleur op als één nummer. Wanneer u een lijst als argument gebruikt en opneemt, wordt deze lijst vervolgens intern geconverteerd met behulp van het kleurenschema van App Inventor en opgeslagen als een getal. Als u de getallen voor de kleuren wist, kunt u zelfs opgeven welke kleur u wilt dat iets heeft door de eigenschap Color in te stellen op een specifiek getal. Als je een grafiek van kleuren naar getallen wilt zien, kijk dan op deze pagina.make color

basiskleur

Dit is een basis kleurblok. Het heeft een kleine vierkante vorm en heeft een kleur in het midden die de kleur vertegenwoordigt die intern in dit blok is opgeslagen.

Als je op de kleur in het midden klikt, verschijnt er een pop-up op het scherm met een tabel van 70 kleuren waaruit je kunt kiezen. Als u op een nieuwe kleur klikt, verandert de huidige kleur van uw basiskleurblok.

Elk basiskleurblok dat u van de kleurenlade naar het scherm Blokkeneditor sleept, geeft een tabel met dezelfde kleuren weer wanneer erop wordt geklikt.

Kleur maken

make color neemt een lijst van 3 of 4 nummers in zich op. Deze getallen in deze lijst vertegenwoordigen waarden in een RGB-code. RGB-codes worden gebruikt om kleuren op internet te maken. Een RGB-kleurenkaart is hier beschikbaar. Dit eerste getal in deze lijst vertegenwoordigt de R-waarde van de code. De tweede staat voor de G. De derde staat voor de B. De vierde waarde is optioneel en vertegenwoordigt de alfawaarde of hoe verzadigd de kleur is. De standaard alfawaarde is 100. Experimenteer met verschillende waarden en kijk hoe de kleuren veranderen met behulp van dit blok.

Gespleten kleur

split color doet het tegenovergestelde van . Het neemt een kleur op: een kleurblok, variabele met een kleur of eigenschap van een van de componenten die een kleur vertegenwoordigen en retourneert een lijst met de RGB-waarden in de RGB-code van die kleur.make color

Variabele Blokken

Er zijn vijf hoofdtypen variabele blokken:

- Initialiseer de globale naam naar
- Toevoegen
- set
- Initialiseer de lokale naam naar in (do)
- Initialiseer de lokale naam naar in (return)

Initialiseer de globale naam naar

Dit blok wordt gebruikt om globale variabelen te maken. Het neemt elk type waarde als argument op. Als u op naam klikt, verandert de naam van deze globale variabele. Globale variabelen worden gebruikt in alle procedures of gebeurtenissen, dus dit blok staat op zichzelf.

Algemene variabelen kunnen worden gewijzigd terwijl een app actief is en kunnen worden geraadpleegd en gewijzigd vanuit elk deel van de app, zelfs binnen procedures en gebeurtenishandlers. U kunt dit blok op elk moment hernoemen en alle bijbehorende blokken die naar de oude naam verwijzen, worden automatisch bijgewerkt.

Toevoegen

Dit blok biedt een manier om alle variabelen op te halen die u mogelijk hebt gemaakt.

set

Dit blok volgt dezelfde regels als . Alleen variabelen binnen het bereik zijn beschikbaar in de vervolgkeuzelijst. Zodra een variabele v is geselecteerd, kunt u een blok toevoegen om v een nieuwe waarde te geven.get

Initialiseer de lokale naam naar - in (do)

Dit blok is een mutator waarmee u nieuwe variabelen kunt maken die alleen worden gebruikt in de procedure die u uitvoert in het DO-gedeelte van het blok. Op deze manier beginnen alle variabelen in deze procedure elke keer dat de procedure wordt uitgevoerd met dezelfde waarde. LET OP: Dit blok wijkt af van het hieronder beschreven blok omdat het een DO-blok is. U kunt er verklaringen aan toevoegen. Verklaringen doen dingen. Daarom heeft dit blok binnenin ruimte om statement blokken te bevestigen.

U kunt de variabelen in dit blok op elk gewenst moment hernoemen en alle overeenkomstige blokken elders in uw programma die naar de oude naam verwijzen, worden automatisch bijgewerkt

Initialiseer de lokale naam naar - in (return)

Dit blok is een mutator waarmee u nieuwe variabelen kunt maken die alleen worden gebruikt in de procedure die u uitvoert in het RETURN-gedeelte van het blok. Op deze manier beginnen alle variabelen in deze procedure elke keer dat de procedure wordt uitgevoerd met dezelfde waarde. LET OP: Dit blok verschilt van het hierboven beschreven blok omdat het een RETURN-blok is. U kunt er expressies aan koppelen. Expressies retourneren een waarde. Daarom heeft dit blok een stopcontact voor het inpluggen van uitingen.

U kunt de variabelen in dit blok op elk gewenst moment hernoemen en alle overeenkomstige blokken elders in uw programma die naar de oude naam verwijzen, worden automatisch bijgewerkt

Procedureblokken

Een procedure is een opeenvolging van blokken of code die is opgeslagen onder een naam, de naam van uw procedureblok. In plaats van steeds dezelfde lange reeks blokken samen te stellen, kunt u een procedure maken en het procedureblok gewoon aanroepen wanneer u wilt dat uw reeks blokken wordt uitgevoerd. In de informatica kan een procedure ook een functie of een methode worden genoemd.

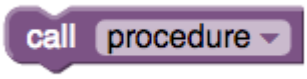
- procedure doen
- Resultaat van de procedure

procedure doen

Verzamelt een reeks blokken samen in een groep. U kunt dan de volgorde van blokken herhaaldelijk gebruiken door de procedure aan te roepen. Als de procedure argumenten heeft, specificeert u de argumenten met behulp van de mutatorknop van het blok. Als u op het blauwe plusteken klikt, kunt u extra argumenten in de procedure slepen.

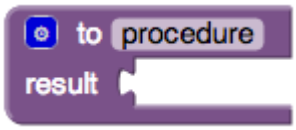
Wanneer u een nieuw procedureblok maakt, kiest App Inventor automatisch een unieke naam. Klik op de naam en typ om deze te wijzigen. Procedurenamen in een app moeten uniek zijn. Met App Inventor kunt u geen twee procedures op hetzelfde scherm met dezelfde naam definiëren. U kunt de naam van een procedure op elk gewenst moment wijzigen terwijl u de app bouwt, door het label in het blok te wijzigen. App Inventor wijzigt automatisch de naam van de bijbehorende oproepblokken.

Java-keywords kunnen niet worden gebruikt als procedurenamen. Hier is een lijst met trefwoorden.

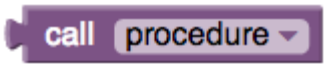


Wanneer u een procedure maakt, genereert App Inventor automatisch een gespreksblok en plaatst dit in de lade Procedures. Je gebruikt het oproepblok om de procedure aan te roepen.

Resultaat van de procedure



Hetzelfde als een procedure blokkeren, maar het aanroepen van deze procedure retourneert een resultaat.



Na het aanmaken van deze procedure wordt er een oproepblok aangemaakt dat moet worden aangesloten. Dit komt omdat het resultaat van het uitvoeren van deze procedure wordt geretourneerd in dat oproepblok en de waarde wordt doorgegeven aan het blok dat op de stekker is aangesloten.