



**Realizzato grazie al contributo  
della Fondazione CRT -  
Torino**

# **Pensiero computazionale**



# Pensiero computazionale

## Apprendimento cognitivo

**L'apprendimento tradizionale** impiega quattro importanti strategie per promuovere la competenza

**Modelling:** l'apprendista osserva ed imita il maestro che dimostra come fare;

**Coaching:** il maestro assiste continuamente secondo le necessità: dirige l'attenzione su un aspetto, dà feedback, agevola il lavoro

**Scaffolding:** è un aspetto particolare del coaching: il maestro fornisce un appoggio all'apprendista, uno stimolo, pre-imposta il lavoro, ecc.

**Fading:** il maestro elimina gradualmente il supporto, in modo da dare a chi apprende uno spazio progressivamente maggiore di responsabilità

**L'apprendistato cognitivo** pone maggiore attenzione alla dimensione metacognitiva, agli aspetti del controllo, ed alla variazione dei contesti di applicazione. Si introducono altre strategie:

**Articolazione:** si incoraggiano gli studenti a verbalizzare la loro esperienza

**Riflessione:** si favorisce il confronto con gli esperti

**Esplorazione:** si spinge a porre e risolvere problemi in forma nuova.

# Pensiero computazionale

## **Che cos'è il pensiero computazionale?**

Non esiste una definizione universalmente condivisa.

Quella che sembra mettere d'accordo il maggior numero di esperti è la definizione formulata dalla dottoressa Jeannette Wing, direttrice del Dipartimento di Informatica della Carnegie Mellon University, secondo cui “è il processo mentale che sta alla base della formulazione dei problemi e delle loro soluzioni così che le soluzioni siano rappresentate in una forma che può essere implementata in maniera efficace da un elaboratore di informazioni sia esso umano o artificiale”.

Ovvero è lo sforzo che un individuo deve mettere in atto per fornire a un altro individuo o macchina tutte e sole le “istruzioni” necessarie affinché questi eseguendole siano in grado di portare a termine il compito dato.

# Pensiero computazionale

Sono in molti oggi a credere che il pensiero computazionale costituisca la **quarta abilità di base** oltre a saper leggere, scrivere e fare di calcolo.

Papert, il padre della teoria dell'apprendimento nota come **costruzionismo**, sostiene che **la mente umana per poter imparare bene ha bisogno di creare artefatti**, ovvero rappresentazioni reali del mondo con cui interagisce.

E il computer, secondo Papert, è un ottimo strumento didattico poiché, grazie alla programmazione, può creare questi artefatti.

# Pensiero computazionale

## Perché è importante il pensiero computazionale?

Nonostante il termine “computazionale” possa indurre a pensare che il pensiero computazionale sia un’abilità utile solo a chi ha fatto dell’informatica la propria professione, si tratta di una **skill fondamentale che tutti dovrebbero possedere**, in particolare quei giovani che desiderano non farsi sfuggire le opportunità che il futuro porrà loro dinanzi nei prossimi anni. È ormai universalmente riconosciuto che per riuscire bene nel proprio futuro professionale i giovani dovranno “imparare a imparare” e non limitarsi a fornire risposte preconfezionate: in questa direzione si muovono le raccomandazioni dell’Unione Europea in materia di istruzione che sono state recepite dal MIUR anche con l’introduzione della programmazione nelle scuole a partire dalla primaria

Ciò che i nostri studenti imparano oggi, tra cinque anni sarà probabilmente obsoleto: questo vale in maniera particolare per le discipline tecniche, ma anche per le altre discipline. Inoltre, è davvero difficile pensare che gli studenti di oggi, che domani troveranno un lavoro, saranno in grado di mantenerlo fino alla fine della propria carriera lavorativa, come accadeva in passato ai loro padri o nonni: è molto più probabile che i giovani saranno chiamati a cambiare lavoro piuttosto frequentemente, imparando a destreggiarsi anche nell’ambito di differenti discipline.

Anche per quanto riguarda le nuove professioni: i lavori oggi più richiesti solo sei anni fa non esistevano neppure.

Proviamo a immaginare quante saranno le discipline che nasceranno nei prossimi anni senza che neppure esistano dei percorsi strutturati per poterle accostare.

La vera sfida per i nostri giovani sarà di porsi in un atteggiamento di lifelong learning e acquisire quelle abilità che consentiranno loro di sviluppare un’attitudine mentale utile ad affrontare problemi di ogni ordine e grado.

# Pensiero computazionale

Il pensiero computazionale è la capacità di elaborare concetti e problemi in forma algoritmica e nello specifico:

Saper analizzare un problema

Saper individuare i dati del problema

Saper identificare, analizzare, implementare e verificare le possibili soluzioni con un'efficace ed efficiente combinazione di passi e risorse (avendo come obiettivo la ricerca della soluzione migliore secondo tali criteri)

Saper codificare, rappresentare i dati del problema tramite opportune astrazioni

Saper organizzare i dati in base a criteri logici

Saper automatizzare la risoluzione del problema definendo una soluzione algoritmica, consistente in una sequenza accuratamente descritta di passi, ognuno dei quali appartenente ad un catalogo ben definito di operazioni di base

Saper generalizzare un problema per poterlo trasferire ad un ampio spettro di altri problemi

Saper operare per artefatti e modelli che si possono utilizzare in diversi contesti.

# Pensiero computazionale

## Qual è il legame tra computer, informatica e pensiero computazionale?

Così come l'invenzione della stampa ha facilitato la diffusione dell'alfabetizzazione, così oggi la programmazione e i computer facilitano l'acquisizione e la diffusione del pensiero computazionale.

Il pensiero computazionale prende a prestito concetti e strumenti propri dell'informatica per trovare soluzioni innovative e creative ai problemi di ogni giorno.

Questo non significa che gli esseri umani devono imparare a pensare come i computer: il pensiero computazionale è il modo in cui gli esseri umani insegnano ai computer a risolvere i problemi e non viceversa. **I computer sono stupidi e noiosi** e solo grazie agli esseri umani possono diventare strumenti utili e interessanti: l'unico limite a quello che i computer sono in grado di fare è costituito dalla nostra **creatività e immaginazione**.

I processi mentali tipici del pensiero computazionale sono favoriti dall'approccio alla risoluzione dei problemi che viene messo in atto da coloro che sviluppano programmi per il computer: in altre parole il pensiero computazionale è quello che adotta un informatico quando affronta un problema. Esercitare il pensiero computazionale significa quindi molto di più che saper scrivere righe di codice.

# Pensiero computazionale

Pratiche di pensiero computazionale:

**Essere incrementali:** la progettazione è un processo adattativo dove la pianificazione può cambiare man mano che ci si avvicina alla soluzione del problema.

**Testare e debuggare:** individuare problemi ed errori e correggerli.

**Riusare** (pattern recognition): riconoscere come alcune parti di soluzione possono essere riusate nella stesso modo o riapplicate a problemi simili.

**Remixare** (copiare per migliorare): grazie alla rete e all'ampia disponibilità di lavori di altri autori, è possibile prendere spunto da idee e codice per costruire cose più complesse di quelle che si sarebbero potute realizzare per conto proprio, dando un'ulteriore spinta alla propria creatività.

**Astrarre:** è il processo di riduzione della complessità, per far emergere l'idea principale mantenendo solo alcuni aspetti e tralasciandone altri.

**Modularizzare** (scomporre): è il processo che consente di scomporre un problema complesso in problemi più semplici, per cui risolvendo i problemi più semplici si risolve anche il problema complesso.

Attitudini di pensiero computazionale:

**Esprimere se stessi:** una persona dotata di pensiero computazionale vede nella tecnologia uno strumento per esprimere se stessi, la propria creatività e dire qualcosa di sé agli altri.

**Essere connessi:** saper comunicare e lavorare con gli altri per raggiungere un obiettivo o una soluzione condivisa.

**Porre domande:** saper sviluppare una mente vigile grazie alla quale è sempre viva la domanda di come un oggetto incontrato nel mondo reale possa funzionare.

# Pensiero computazionale

Concetti di pensiero computazionale:

**Sequenza:** un'attività può essere espressa attraverso una serie consecutiva di singoli step o istruzioni.

**Ciclo:** è un meccanismo per eseguire più volte la medesima sequenza in maniera iterativa.

**Evento:** il verificarsi di un'azione causa lo scatenarsi di un'altra azione.

**Parallelismo:** significa eseguire sequenze di istruzioni differenti allo stesso tempo.

**Condizione:** è la possibilità di prendere decisioni sulla base del verificarsi di determinate situazioni.

**Operatore:** fornisce supporto per la manipolazione di numeri e stringhe di caratteri.

**Dati:** sono valori che possono essere salvati, recuperati e modificati durante l'esecuzione di un programma.

# Pensiero computazionale

Il testimone di Papert è stato raccolto da Mitchel Resnick responsabile del Lifelong Kindergarten del MIT MediaLab che con i suoi collaboratori ha realizzato un frame work per l'insegnamento del pensiero computazionale e la valutazione dell'apprendimento che si fonda sulla convinzione che **i bambini possano acquisire il pensiero computazionale programmando storie interattive e videogiochi** (gli artefatti di cui parlava Papert).

Il lavoro di questi anni di Resnick e dei suoi collaboratori ha portato alla nascita di **Scratch**, un ambiente di programmazione visuale che consente ai ragazzi di creare in maniera semplice e intuitiva le proprie storie animate, giochi e simulazioni: oggi Scratch conta una community di giovani sviluppatori estesa in tutto il mondo ed è di fatto lo strumento di riferimento per insegnare ai bambini il pensiero computazionale attraverso la programmazione.

# Pensiero computazionale

Il software **Scratch** è un «tool» di programmazione visuale (il codice del programma non deve essere digitato) ideato al Mit di Boston.

**Blockley** è un ambiente di programmazione grafico, l'utente sposta dei blocchi, simili a tasselli di un puzzle, per realizzare le proprie applicazioni.

Ricorda altri linguaggi di programmazione visuale, infatti storicamente la realizzazione di Blockley è stata influenzata da **App Inventor**, App Inventor è stato influenzato da **Scratch** il quale è stato influenzato da **StarLogo** e **Etoys** .

Il progenitore di tutti questi linguaggi di programmazione è stato il Logo, creato da Seymour Papert insieme a Jean Piaget, fautore della teoria dell'apprendimento costruttivista

# Pensiero computazionale

- Dare i comandi alle macchine
- Preparare serie di azioni partendo dall'obiettivo
- 
- Account gmail
- Ai.2.appinventor MIT
- Permetti
- Lingua                      italiano
- Progetti                    avvio                      nome
- Screen 1                    titolo

# Pensiero computazionale

Progettazione				Blocchi
Compon enti	Visualizz azione	Compon enti utilizzati	proprietà	
			colore	
			Immagini	
			testo	

# Pensiero computazionale

Cosa si dovrà vedere alla fine

Screen 1

proprietà

Colore

Posizione schermo

Immagine

multimediale

carica immagine

Immagine proprietà

altezza

larghezza

immagine

carica file

Casella di testo

lato 1 se rinominare da 3 parte

Casella di testo

lato 2 se rinominare da 3 parte

Pulsante

per schiacciare per calcolo area

Etichetta

risultato

Se vogliamo spazi inserire caselle etichette



# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes the MIT App Inventor 2 Beta logo, a menu (Projects - Connect - Build - Help -), and user information (My Projects, Gallery, Guide, Report an Issue, English - grazmagn@gmail.com -). Below this is a green header with the project name 'cod2' and buttons for 'Screen1 -', 'Add Screen ...', and 'Remove Screen'. On the right of the header are 'Designer' and 'Blocks' buttons.

The main workspace is divided into four panels:

- Palette:** Contains categories for 'User Interface' (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, WebViewer), 'Layout', 'Media', 'Drawing and Animation', 'Sensors', 'Social', 'Storage', 'Connectivity', and 'LEGO® MINDSTORMS®'.
- Viewer:** Shows a mobile app preview. The app has a status bar at the top with 'Home', signal strength, Wi-Fi, and the time '9:48'. The main content area features a home icon, the text 'LCA @Coding', a 'Calcola' button with a plus icon, and an 'Informazioni' button with an info icon. The bottom of the viewer shows a standard Android navigation bar.
- Components:** Lists the components on the screen: 'Screen1' (expanded) containing 'VerticalScrollArrangemen', 'Label1', 'Image1', 'lb\_sottotitolo', 'bn\_calcola', 'lb\_pulsante1', 'bn\_info', and 'lb\_pulsante2'. Below the list are 'Rename' and 'Delete' buttons.
- Properties:** Shows the properties for 'Screen1', including 'AboutScreen', 'AlignHorizontal' (Center: 3), 'AlignVertical' (Top: 1), 'AppName' (LCA @Coding), 'BackgroundColor' (White), 'BackgroundImage' (None...), 'CloseScreenAnimation' (Default), 'Icon' (ico.png...), 'OpenScreenAnimation' (Default), 'ScreenOrientation' (Unspecified), 'Scrollable' (unchecked), 'ShowStatusBar' (checked), 'Sizing' (Responsive), 'Title' (Home), and 'TitleVisible'.

# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. At the top, the navigation bar includes 'MIT App Inventor 2 Beta', 'Projects -', 'Connect -', 'Build -', 'Help -', 'My Projects', 'Gallery', 'Guide', 'Report an Issue', 'English -', and 'grazmagn@gmail.com -'. Below this is a green header with 'cod2', 'calcola -', 'Add Screen ...', 'Remove Screen', 'Designer', and 'Blocks'.

The interface is divided into four main panels:

- Palette:** Contains categories for 'User Interface' (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, WebViewer), 'Layout', 'Media', 'Drawing and Animation', 'Sensors', 'Social', 'Storage', 'Connectivity', and 'LEGO® MINDSTORMS®'.
- Viewer:** Shows a mobile app preview titled 'calcola'. It features a title bar with 'calcola', a status bar with signal, Wi-Fi, and battery icons, and a time of 9:48. The app content includes a title 'Calcola', four input fields labeled 'Laterizi', 'Cis', 'Isolanti', and 'Acciaio', each with an 'add items...' button and a numeric field. At the bottom, there are two 'GWP' (Google Web Player) buttons and a refresh icon.
- Components:** A tree view showing the app's structure: 'calcola' contains 'VsA\_ipo1' (Label9, lb\_titolo), 'VA\_ipo1' (HorizontalArran..., lb\_tipo1, sp\_1\_1, Label17, tb1\_1), another 'HorizontalArran...' (lb\_tipo2, sp\_1\_2, Label18, tb1\_2), and a third 'HorizontalArran...' (lb\_tipo3, sp\_1\_3, tb1\_3).
- Properties:** Shows the properties for the selected 'HA\_ris\_1' component, including 'AlignHorizontal' (Center: 3), 'AlignVertical' (Center: 2), 'BackgroundColor' (White), 'Height' (Automatic...), 'Width' (Fill parent...), 'Image' (None...), and 'Visible' (checked).

At the bottom right, a 'Media' panel lists various assets: 1.png, 2.png, Chart.min.js, aggiorna.png, confronta.png, grafico.png, home.png, ico.png, info.png, and pie.html.

# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes the MIT App Inventor logo, the text "MIT App Inventor 2 Beta", and menu items: "Projects", "Connect", "Build", "Help", "My Projects", "Gallery", "Guide", "Report an Issue", "English", and "grazmagn@gmail.com". Below the navigation bar, the project name "cod2" is shown, along with "Screen1" and buttons for "Add Screen..." and "Remove Screen". The interface is divided into three main sections: "Blocks", "Viewer", and "Media".

**Blocks:** A vertical list of component categories is shown on the left. Under "Built-in", there are categories for Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. Under "Screen1", there is a "VerticalScrollArrangen" component with sub-components: "Label1", "Image1", "lb\_sottotitolo", "bn\_calcola", "lb\_pulsante1", "bn\_info", and "lb\_pulsante2". At the bottom of the list, there is a search bar with "Any component" and "Rename" and "Delete" buttons.

**Viewer:** The central workspace shows two event blocks. The first block is a "when bn\_calcola Click" event with a "do" block containing "open another screen" with "screenName" set to "calcola". The second block is a "when bn\_info Click" event with a "do" block containing "open another screen" with "screenName" set to "info". A green trash can icon is visible in the top right corner of the viewer area.

**Media:** A list of media files is shown at the bottom left, including "1.png", "2.png", "Chart.min.js", "aggioma.png", "confronta.png", "grafico.png", "home.png", "ico.png", and "info.png".

At the bottom of the viewer area, there is a "Show Warnings" button with two warning icons and the number "0" next to each.

# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes the MIT App Inventor logo, the text "MIT App Inventor 2 Beta", and menu items: "Projects", "Connect", "Build", "Help", "My Projects", "Gallery", "Guide", "Report an Issue", "English", and "grazmagn@gmail.com". Below the navigation bar, the interface is divided into three main sections: "Blocks", "Viewer", and "Media".

The "Blocks" section on the left contains a tree view of components. Under "Built-in", there are categories for Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. Under "info", there is a "VerticalScrollArrangement" component with sub-components "lb\_spazio", "image1", "Label1", and "lb\_info". There is also an "Any component" category. At the bottom of the "Blocks" section are "Rename" and "Delete" buttons.

The "Viewer" section on the right shows a code block for the "info" component's ".Initialize" event. The code block is a "do" block containing a "set lb\_info . Text to" block followed by a "join" block. The "join" block contains the following text:

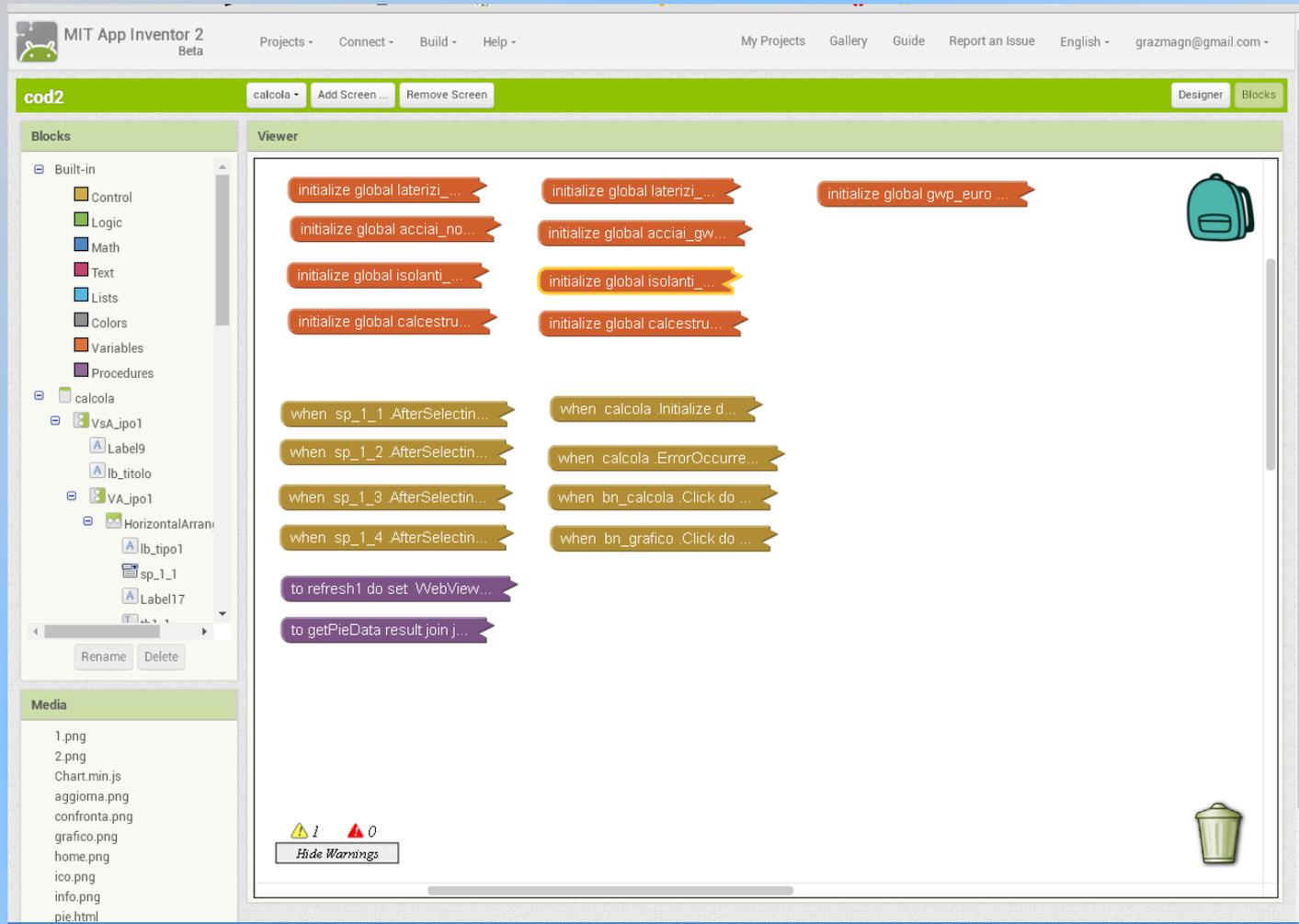
```
" L'applicazione "LCA@Coding", sviluppata a scopo didattico per avvicinare gli studenti alla pro  
- la valutazione del GWP (potenziale di riscaldamento globale di un gas ad effetto serra rispet  
- la valutazione dei costi indiretti (esternalita' ambientali) secondo i fattori di monetizzazione ri  
Ulteriori specifiche sono da richiedersi a: Associazione ETA.\n  
Tutti i diritti riservati. "
```

The "Media" section at the bottom left lists various image and document files: "1.png", "2.png", "Chart.min.js", "aggiorna.png", "confronta.png", "grafico.png", "home.png", "ico.png", "info.png", and "pie.html".

At the bottom of the "Viewer" section, there are two warning icons (a yellow triangle and a red triangle) with a "0" next to each, and a "Show Warnings" button. A trash can icon is also visible in the bottom right corner of the viewer area.

# Pensiero computazionale

## App inventor



# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes 'MIT App Inventor 2 Beta', 'Projects', 'Connect', 'Build', 'Help', 'My Projects', 'Gallery', 'Guide', 'Report an Issue', 'English', and 'grazmagn@gmail.com'. The main workspace is titled 'cod2' and contains a 'Designer' tab and a 'Blocks' tab. The 'Blocks' panel on the left shows a tree view of the project structure, including a 'calcola' screen with several sub-screens and a 'HorizontalArran...' widget. The 'Viewer' panel on the right shows a visual representation of the code blocks. The code includes several 'initialize global' blocks for variables like 'acciai\_no...', 'laterizi\_...', 'isolanti\_...', and 'calcestru...'. It also features several 'when' blocks for handling events such as 'AfterSelectin...', 'Initialize d...', 'ErrorOccurre...', 'Click do ...', and 'AfterRefresh1 do set WebView...'. A 'to refresh1 do set WebView...' block is also present. At the bottom, a 'make a list' block is connected to a list of items: '1-mattoni faccia vista-p.s.1800', '2-mattoni in calcestruzzo leggero-p.s.600', '3-prodotti prefabbricati in calcestruzzo-p.s.2000', '4-tegole in cemento-p.s.2150', '5-tegole in cotto-p.s.2000', and '6-mattoni-p.s.575'. The interface also includes a 'Media' panel with various image and HTML files, and a 'Designer' tab with a 'Blocks' button.

# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes 'MIT App Inventor 2 Beta', 'Projects', 'Connect', 'Build', 'Help', 'My Projects', 'Gallery', 'Guide', 'Report an Issue', 'English', and 'grazmagn@gmail.com'. The main workspace is divided into three sections: 'Blocks', 'Viewer', and 'Media'.

**Blocks Panel:** Shows a hierarchical tree of components. Under 'Built-in', there are categories for Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. A project-specific folder 'calcola' contains a sub-folder 'VsA\_jpo1' with components 'Label9', 'lb\_titolo', and 'VA\_jpo1'. The 'VA\_jpo1' folder contains 'HorizontalArran', 'lb\_tipo1', 'sp\_1\_1', and 'Label17'.

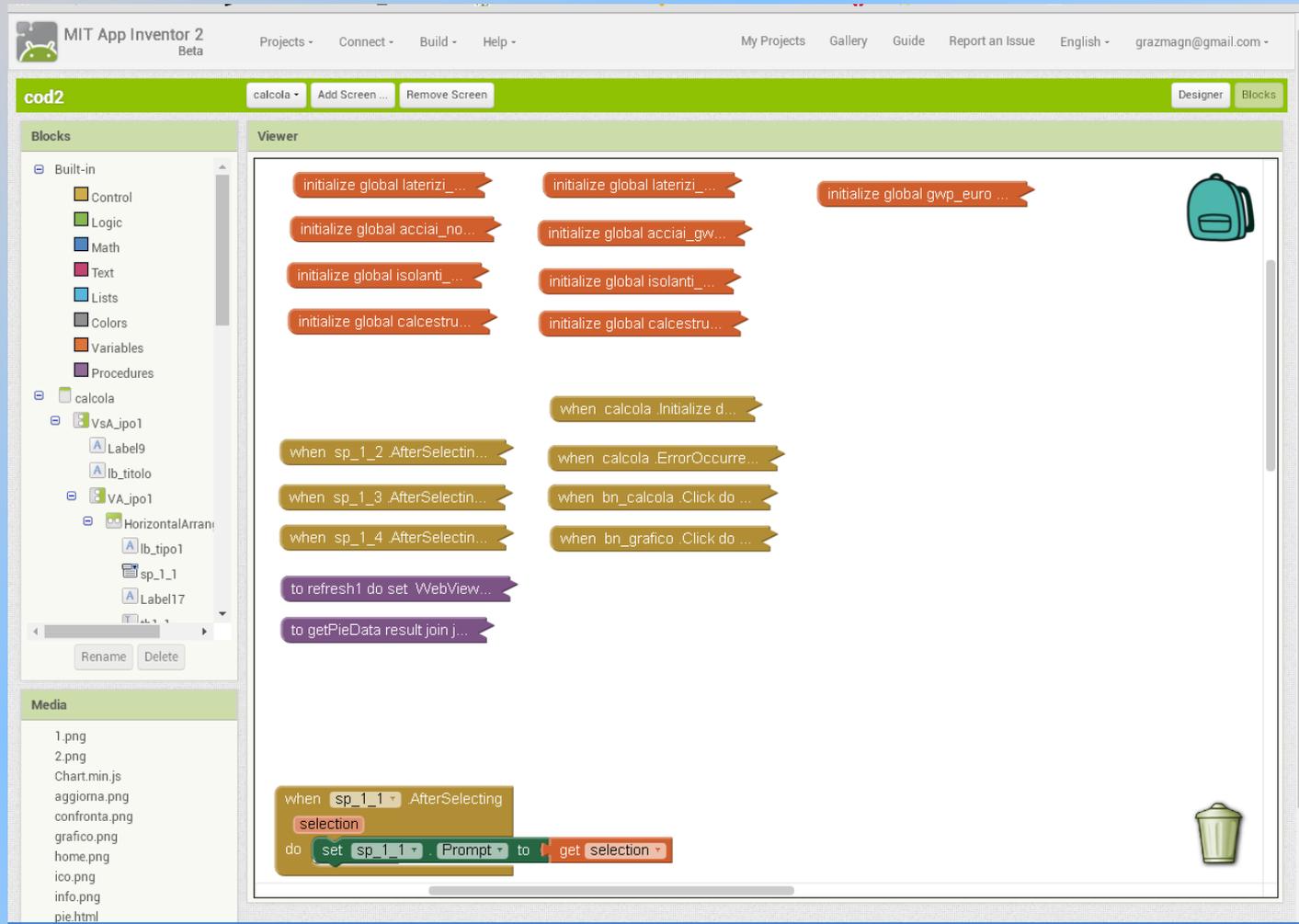
**Viewer Panel:** Displays the visual representation of the app's code blocks. It includes several 'initialize global' blocks for variables like 'laterizi\_gwp', 'acciai\_no...', 'isolanti...', and 'calcestru...'. There are also 'when' blocks for events such as 'AfterSelectin...', 'Initialize d...', 'ErrorOccurre...', and 'Click do...'. A 'to refresh1 do set WebView...' block and a 'to getPieData result join j...' block are also visible.

**Media Panel:** Lists various image and audio files, including '1.png', '2.png', 'Chart.min.js', 'aggiorna.png', 'confronta.png', 'grafico.png', 'home.png', 'ico.png', 'info.png', and 'pie.html'.

The main workspace shows a visual representation of the app's interface with a green header bar containing 'cod2', 'calcola', 'Add Screen...', and 'Remove Screen'. The interface includes a 'Designer' button and a 'Blocks' button. The visual representation shows a grid of code blocks and a trash icon.

# Pensiero computazionale

## App inventor



# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. At the top, the browser address bar shows the URL `http://appinventor.mit.edu`. The page header includes the MIT App Inventor 2 Beta logo, navigation links (Projects, Connect, Build, Help), and user information (My Projects, Gallery, Guide, Report an Issue, English, and email `grazmagn@gmail.com`).

The main workspace is titled "cod2" and is divided into two panes: "Designer" and "Blocks". The "Designer" pane shows a visual programming workspace with several blocks arranged in a grid. The "Blocks" pane on the left contains a palette of built-in blocks, including Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. The "Media" pane at the bottom left lists various image and audio files.

The visual programming workspace contains the following blocks:

- Initialize global laterizi\_...
- Initialize global acciai\_no...
- Initialize global isolanti\_...
- Initialize global calcestru\_...
- Initialize global gwp\_euro ...
- Initialize global acciai\_gw...
- Initialize global isolanti\_...
- Initialize global calcestru\_...
- When sp\_1\_1\_AfterSelectin...
- When sp\_1\_2\_AfterSelectin...
- When sp\_1\_3\_AfterSelectin...
- When sp\_1\_4\_AfterSelectin...
- When calcola Initialize d...
- When calcola\_ErrorOccurre...
- When bn\_calcola\_Click do ...
- When bn\_grafico\_Click do ...
- To getPieData result join j...
- To refresh1
- Do set WebViewer1 WebViewString to call getPieData
- Call WebViewer1 GoHome

The workspace also features a trash can icon in the bottom right corner and a backpack icon in the top right corner.

# Pensiero computazionale

## App inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes 'Projects', 'Connect', 'Build', and 'Help'. The main workspace is divided into three sections: 'Blocks', 'Viewer', and 'Media'.

**Blocks Panel:** Lists various categories of blocks including Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. A project named 'calcola' is open, showing a hierarchy of components like 'VsA\_ipo1', 'Label9', 'lb\_titolo', 'VA\_ipo1', and 'HorizontalArran'.

**Code Editor:** Shows a sequence of logic blocks for the 'when bn\_calcola.Click' event:

- Call `calcola.HideKeyboard`
- Set `lb_ris1_GWP.Text` to 0
- Set `lb_ris1_EURO.Text` to 0
- Four 'if' blocks, each checking if a spinner's `SelectionIndex` is 1 and the corresponding text box (`tb1_1` to `tb1_4`) is empty. If true, the result label (`lb_risultato1_1_GWP` to `lb_risultato1_4_GWP`) is set to 0.
- A final 'if' block checking if `sp_1_1.SelectionIndex` is greater than 1 and the text box is not empty. If true, it sets `lb_valoreGWP_1_1.Text` to `select list item list index` from `sp_1_1.SelectionIndex`.

**Media Panel:** Lists various image and document files such as '1.png', 'Chart.min.js', 'aggiorna.png', 'confronta.png', 'grafico.png', 'home.png', 'ico.png', 'info.png', and 'pie.html'.

# Pensiero computazionale

App inventor

# Pensiero computazionale

App inventor