Słowniczek podstawowych bloków programu Scratch 2.0

www.mistrzowiekodowania.pl

ORGANIZATORZY

PATRONI

Słowniczek podstawowych bloków programu Scratch 2.0

Scratch to skryptowy, wizualny język programowania. Budujemy w nim skrypty układając je z bloków reprezentujących polecenia, instrukcje sterujące, wyrażenia itd. Zestaw bloków w wersji 2.0 podzielony jest na 10 kategorii (w wersji 1.4 było 8 kategorii), bloki każdej z nich oznaczone są innym kolorem. Niniejszy słowniczek nie stanowi opisu wszystkich bloków, jedynie najważniejszych oraz wykorzystywanych w scenariuszach zajęć przeznaczonych dla uczniów klas IV-VI szkół podstawowych opracowanych w ramach projektu „Mistrzowie kodowania”. W wersji 2.0 wprowadzono bardzo ważną kategorię **Więcej bloków** umożliwiającą definiowanie własnych bloków. Nie została ona tutaj opisana, ponieważ pierwszych osiem scenariuszy nie przewiduje tworzenia własnych bloków. Będzie opisana przy następnych scenariuszach.

Każdy duszek, a także scena, może posiadać własne skrypty. Zestaw bloczków, z których można budować skrypty dla sceny jest nieco inny niż dla duszków, np. kategoria **Ruch** jest dla sceny pusta, ponieważ sceny nie można przesuwać ani obracać. Najpierw zostaną omówione poszczególne kategorie bloczków dla duszków, a następnie podane podstawowe informacje o bloczkach dostępnych dla sceny.

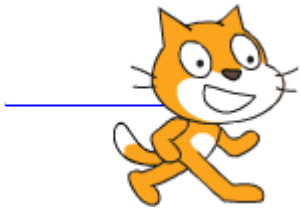
RUCH

W kategorii **Ruch** zgromadzone są bloczki związane z ruchem duszków, a więc ich przesuwaniem, obracaniem oraz odczytywaniem aktualnych danych związanych z położeniem duszka. Są to podstawowe bloczki bardzo często używane. Poniżej znajduje się opis wszystkich bloczków tej kategorii.

1. przesuń o 10 kroków

Przesuwa duszka o określoną odległość zgodnie z aktualnym kierunkiem oraz aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Domyślną wartość możemy edytować. Jeśli chcemy przesunąć duszka do tyłu należy podać wartość ujemną.

Przykład:

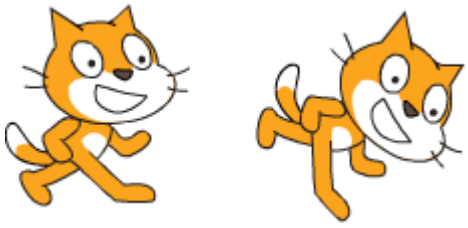


---

2. obróć o 15 stopni

Obraca duszka w prawo (zgodnie z ruchem wskazówek zegara) o określoną liczbę stopni. Domyślną wartość możemy edytować.

Przykład:



---

3. obróć o 15 stopni

Obraca duszka w lewo (przeciwnie do ruchu wskazówek zegara) o określoną liczbę stopni. Domyślną wartość możemy edytować. Jest równoważne obrotowi w prawo z ujemną wartością.

Przykład:

---

4. ustaw kierunek na 90

Ustawia duszka na określony kierunek. Można z listy wybrać jeden z czterech kierunków (90 stopni – prawo, -90 stopni – lewo, 0 stopni – do góry, 180 stopni – dół) lub wpisać ręcznie określony kąt wyrażony w stopniach.

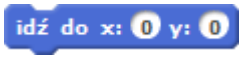
Przykład:

---

5. ustaw w stronę

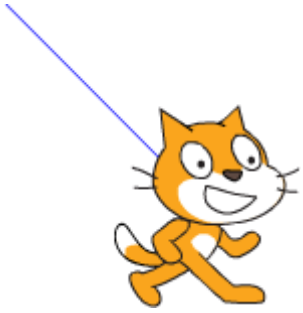
Ustawia duszka w kierunku na kursor myszy lub innego duszka. Obiekt (określonego duszka lub kursor myszy) wybieramy z listy rozwijalnej.

Przykład:

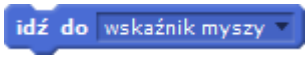
6. 

Przesuwa duszka do punktu o określonych współrzędnych (x,y) zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość).

Przykład:

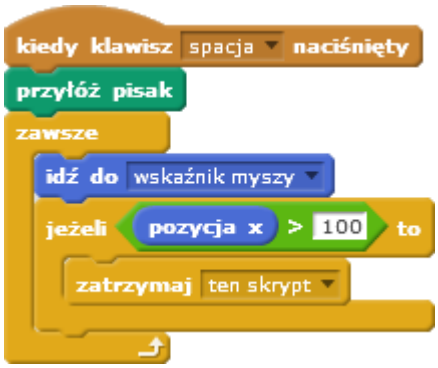


---

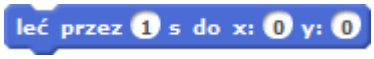
7. 

Przesuwa duszka do pozycji kursora myszy lub innego duszka zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Obiekt (określonego duszka lub kursor myszy) wybieramy z listy rozwijalnej.

Przykład:

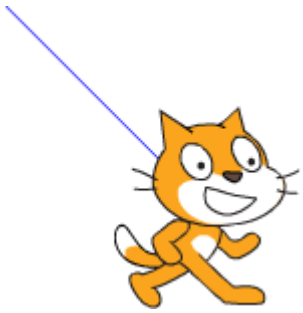


---

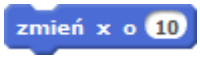
8. 

Analogiczne do polecenia **idź do x y**, ale wykonywane przez określony czas.

Przykład:

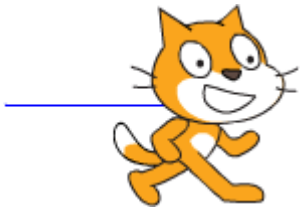


---

9. 

Zmienia współrzędną x-ową duszka o określoną wartość zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Domyślną wartość możemy edytować.

Przykład:

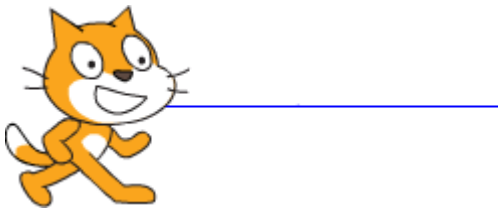


---

10. 

Przesuwa duszka do punktu o określonej współrzędnej x-owej (y-owa współrzędna pozostaje bez zmiany) zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Domyślną wartość możemy edytować.

Przykład:



---

11. 

Zmienia współrzędną y-ową duszka o określoną wartość zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Domyślną wartość możemy edytować.

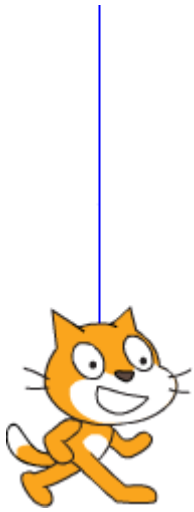
Przykład:

---

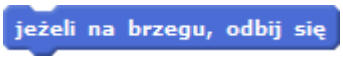
12. 

Przesuwa duszka do punktu o określonej współrzędnej y-owej (x-owa współrzędna pozostaje bez zmiany) zgodnie z aktualnymi ustawieniami pisaka (przyłożony/podniesiony, kolor, grubość). Domyślną wartość możemy edytować.

Przykład:



---

13. 

Zmienia kierunek na przeciwny, jeśli duszek dotyka krawędzi sceny.

Przykład:

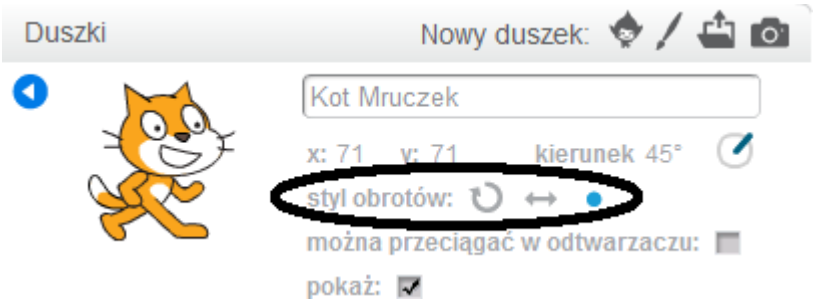
---

14. 

Określa sposób zachowania duszka w zależności od kierunku. Z listy rozwijalnej mamy do wyboru trzy opcje:

- dookoła – duszek patrzy zgodnie ze swoim kierunkiem,
- lewo-prawo – duszek patrzy tylko w lewą lub prawą stronę,
- nie obracaj – duszek nie reaguje na zmianę kierunku.

Niezależnie od sposobu wskazywania kierunku duszek porusza się zgodnie ze swoim kierunkiem. Sposób wskazywania kierunku można także określić edycyjnie.



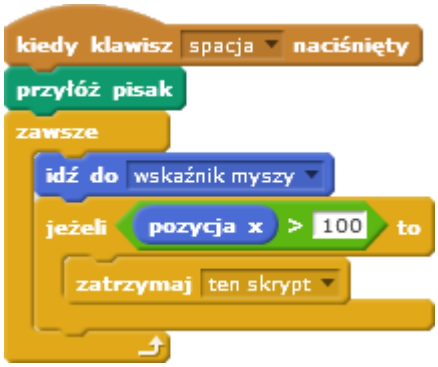
Przykład:

---

15. pozycja x

Aktualna wartość współrzędnej x-owej pozycji duszka – jego środka. Środek można ustawić edytując kostium duszka. Zwracaną wartość można wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie aktualnej współrzędnej na scenie. Przyjmuje wartości od -240 do 240.

Przykład:



---

16. pozycja y

Aktualna wartość współrzędnej y-owej pozycji duszka – jego środka. Środek można ustawić edytując kostium duszka. Zwracaną wartość można wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie aktualnej współrzędnej na scenie. Przyjmuje wartości od -180 do 180.

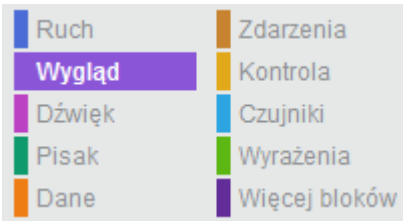
Przykład:

---

17. kierunek

Aktualna wartość kierunku duszka. Można ją wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie aktualnego kierunku na scenie. Przyjmuje wartości od -180 do 180 stopni.

Przykład:



WYGLĄD

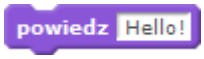
W kategorii **Wygląd** zgromadzone są bloczki związane z wyglądem duszków, a więc kostiumem, wielkością, widocznością, itp.. Niektóre z tych bloczków są bardzo często używane. Poniżej znajduje się opis większości bloczków tej kategorii.

1. 

Przez określony czas wyświetla w dymku tekst podany w polu edycyjnym. Oznacza to wstrzymanie działania skryptu na podaną liczbę sekund. Do pola edycyjnego można także przeciągnąć zmienną lub zbudować w tym miejscu wyrażenie, często do połączenia wyświetlanych informacji wykorzystujemy bloczek **połącz**.

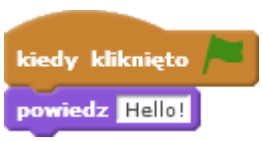
Przykład:

---

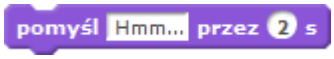
2. 

Wyświetla w dymku tekst podany w polu edycyjnym i nie wstrzymuje działania skryptu. Wykonywane są w związku z tym kolejne instrukcje, co może powodować szybkie zniknięcie dymku. Do pola edycyjnego można także przeciągnąć zmienną lub zbudować w tym miejscu wyrażenie, często do połączenia wyświetlanych informacji wykorzystujemy bloczek **połącz**.

Przykład:



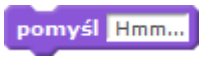
---

3. 

Przez określony czas wyświetla w dymku tekst podany w polu edycyjnym. Oznacza to wstrzymanie działania skryptu na podaną liczbę sekund. Działa analogicznie jak polecenie **powiedz**, inny jest tylko rodzaj wyświetlanej informacji. Do pola edycyjnego można także przeciągnąć zmienną lub zbudować w tym miejscu wyrażenie, często do połączenia wyświetlanych informacji wykorzystujemy bloczek **połącz**.

Przykład:

---

4. 

Wyświetla w dymku tekst podany w polu edycyjnym i nie wstrzymuje działania skryptu. Wykonywane są w związku z tym kolejne instrukcje, co może powodować szybkie zniknięcie wyświetlanej informacji. Działa analogicznie jak polecenie **powiedz**, inny jest tylko rodzaj dymku. Do pola edycyjnego można także przeciągnąć zmienną lub zbudować w tym miejscu wyrażenie, często do połączenia wyświetlanych informacji wykorzystujemy bloczek **połącz**.

Przykład:

---

5. 

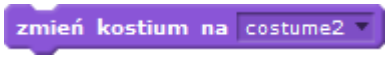
Powoduje wyświetlenie obrazu duszka (aktualnego kostiumu), jeśli duszek jest ukryty. Jeśli duszek jest widoczny nie powoduje żadnego efektu.

---

6. 

Powoduje ukrycie obrazu duszka, jeśli duszek jest widoczny. Jeśli duszek jest niewidoczny nie powoduje żadnego efektu.

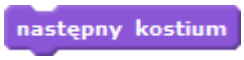
---

7. 

Powoduje zmianę kostiumu duszka na określony, wybrany z listy rozwijalnej. Domyślny duszek (kotek) ma dwa kostiumy.

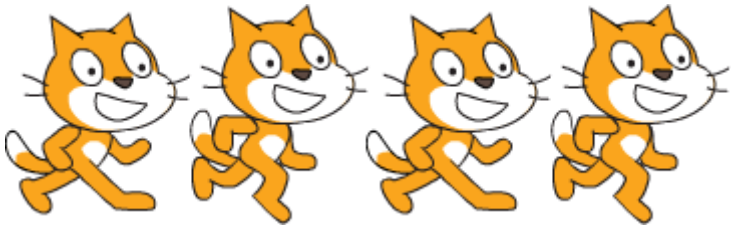
Przykład:

---

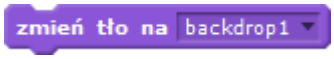
8. 

Powoduje zmianę kostiumu duszka na następny. Następnym kostiumem po ostatnim jest ponownie pierwszy. Klocek ten jest często wykorzystywany do animacji postaci duszka.

Przykład:

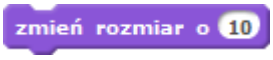


---

9. 

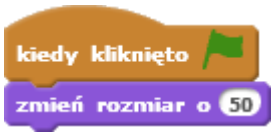
Scena może się składać z wielu scen (domyślnie jest tylko jedno, białe tło). Przy pomocy tego klocka duszek może zmienić tło na wybrane z listy rozwijalnej (o konkretnej nazwie, następne tło, poprzednie tło). Uwaga: klocek niedostępny dla skryptów duszków w wersji 1.4 (dostępny jedynie dla skryptów sceny).

---

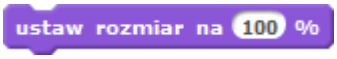
10. 

Powoduje zmianę wielkości duszka (wszystkich kostiumów) o określoną wartość. Jeśli chcemy pomniejszyć wielkość duszka należy podać ujemny argument.

Przykład:

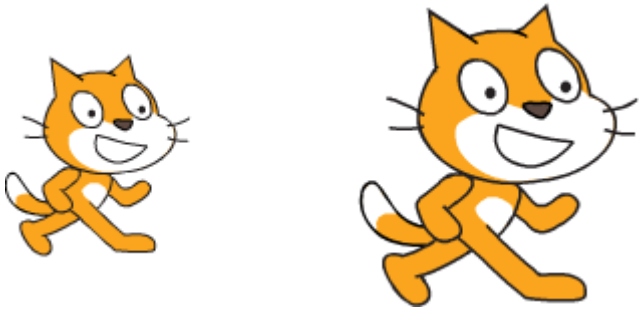


---

11. 

Powoduje zmianę wielkości duszka (wszystkich kostiumów) na określoną wielkość wyrażoną w procentach. Jeśli chcemy przywrócić oryginalną wielkość należy ustawić rozmiar na 100%.

Przykład:



---

12. 

Jeśli duszek jest przykryty przez inne duszki, zostanie wyświetlony na wierzchu.

Przykład:

---

13. 

Aktualny numer kostiumu duszka. Można go wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie aktualnego numeru na scenie.

Przykład:

---

14. nazwa tła

Nazwa aktualnie wyświetlanego tła. Można ją wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie nazwy tła na scenie.

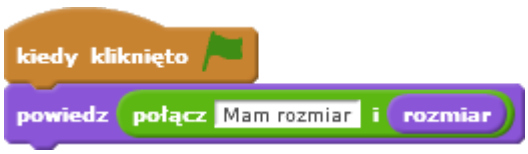
Przykład:

---

15. rozmiar

Aktualny rozmiar duszka wyrażony w procentach. Można go wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie rozmiaru na scenie.

Przykład:



DŹWIĘK

W kategorii **Dźwięk** zgromadzone są bloczki związane z dźwiękami, instrumentami i nutami. Poniżej znajduje się opis jedynie bloczków związanych z odtwarzaniem plików dźwiękowych. Z duszkiem mogą być związane różne dźwięki (przypisane im pliki dźwiękowe), domyślne duszek kot ma przypisany jeden dźwięk – miauczenie. Zmian dźwięków przypisanych duszkowi (usunięcie/dodanie pliku dźwiękowego, nagranie własnego dźwięku) dokonujemy w zakładce **Dźwięki**.

1. 

Powoduje odtworzenie pliku dźwiękowego wybranego z listy rozwijalnej. Kolejne instrukcje skryptu są wykonywane niezależnie od odtwarzanego dźwięku.

Przykład:

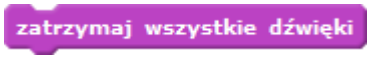
---

2. 

Powoduje odtwarzanie pliku dźwiękowego wybranego z listy rozwijalnej oraz wstrzymanie działania skryptu. Kolejne instrukcje skryptu są wykonywane po zakończeniu odtwarzania pliku dźwiękowego.

Przykład:

---

3. 

Powoduje zatrzymanie wszystkich odtwarzanych aktualnie dźwięków.

PISAK

W kategorii **Pisak** zgromadzone są bloczki związane z pisakiem duszka, jego stanem, kolorem, grubością. W tej grupie bloczków znajduje się również klocek do czyszczenia stworzonych rysunków oraz możliwość odbicia wizerunku duszka jako pieczątki. Poniżej znajduje się opis wszystkich bloczków tej kategorii.

1. 

Powoduje wyczyszczenie rysunków utworzonych przez wszystkie duszki. Nie ma wpływu na tło sceny.

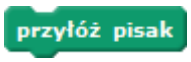
---

2. 

Powoduje przystemplowanie wizerunku duszka (aktualnego kostiumu) jak odbicie pieczątki.

Przykład:

---

3. 

Powoduje opuszczenie pisaka duszka. Wykonywane ruchy będą powodowały rysowanie zgodnie z ustawieniami pisaka (kolor, grubość).

Przykład:

---

4. podnieś pisak

Powoduje podniesienie pisaka duszka. Wykonywane ruchy duszka nie będą powodowały rysowania. Przykład: patrz opis bloczka **przyłóż pisak**.

---

5. ustaw kolor pisaka na

Powoduje wybranie koloru pisaka. Należy kliknąć myszką w kwadracik z kolorem, następnie wybrać kolor kursorem myszy z dostępnych na ekranie. W poniższym przykładzie wybrano kolor zielony klikając w zieloną flagę oraz kolor czerwony w przycisk zatrzymania wykonywania skryptów. W zależności od wybranego koloru zmienia także ustawienia odcienia koloru.

Uwaga: W wersji 1.4 po kliknięciu w kwadracik klocka z kolorem wyświetlana była również paleta kolorów do wyboru. Niestety obecnie w wersji 2.0 nie ma tej możliwości.

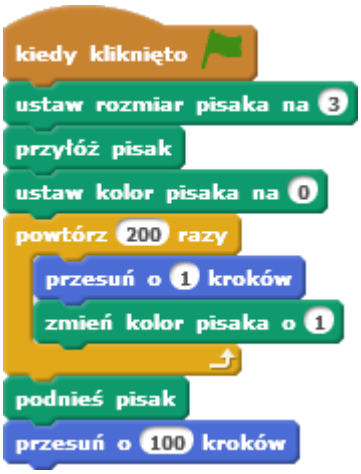
Przykład:

---

6. zmień kolor pisaka o 10

Każdy kolor posiada swoją wartość liczbową. Numery kolorów zmieniają się w zakresie od 0 do 200 przechodząc przez kolory tęczy. Kolor o numerze 0 to czerwony, 70 to zielony, 130 niebieski, a 170 karmazynowy (magenta). Polecenie powoduje zmianę wartości koloru o podany argument.

Przykład:

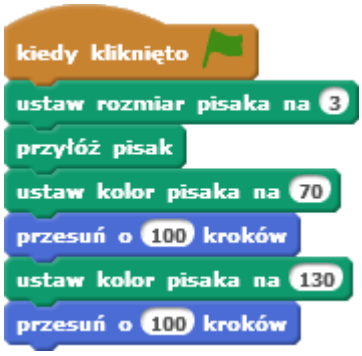


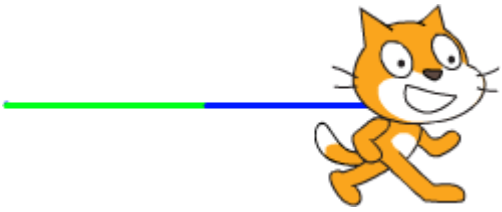
---

7. ustaw kolor pisaka na 0

Każdy kolor posiada swoją wartość liczbową. Numery kolorów zmieniają się w zakresie od 0 do 200 przechodząc przez kolory tęczy. Kolor o numerze 0 to czerwony, 70 to zielony, 130 niebieski, a 170 karmazynowy (magenta). Polecenie powoduje ustawienie koloru zgodnie z podaną wartością liczbową.

Przykład:





---

8. zmień odcień pisaka o 10

Każdy kolor posiada swój odcień określony wartością liczbową w zakresie od 0 do 100. Odcień o wartości 0 jest zbliżony do czerni, odcień o wartości 100 do bieli. Standardową wartością jest 50. Polecenie powoduje zmianę wartości odcienia o podany argument.

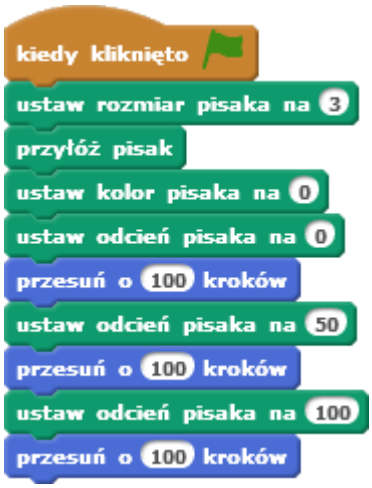
Przykład:

---

9. ustaw odcień pisaka na 50

Każdy kolor posiada swój odcień określony wartością liczbową w zakresie od 0 do 100. Odcień o wartości 0 jest zbliżony do czerni, odcień o wartości 100 do bieli. Standardową wartością jest 50. Polecenie powoduje ustawienie wartości odcienia na podaną wartość.

Przykład:

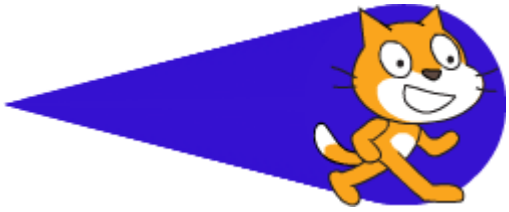


---

10. zmień rozmiar pisaka o 1

Polecenie powoduje zmianę wartości grubości pisaka o podany argument.

Przykład:



---

11. 

Polecenie ustawia grubość pisaka na podaną wartość.

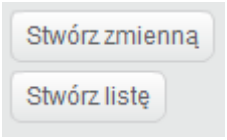
Przykład:

DANE

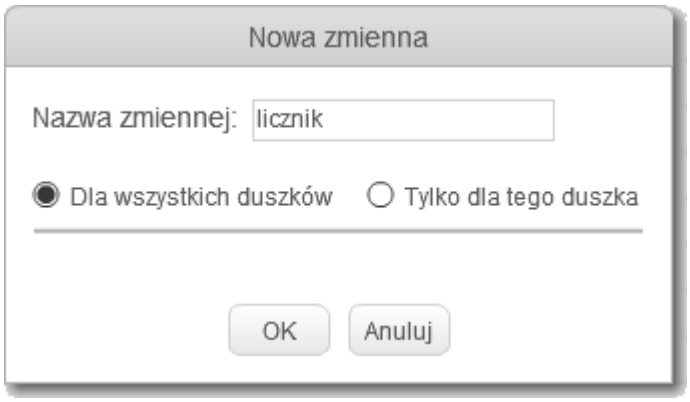
Kategoria **Dane** zawiera przyciski do tworzenia zmiennych. Po utworzeniu własnych zmiennych pojawią się bloczki do ich modyfikowania oraz wykorzystania wartości. Zmienne umożliwiają przechowywanie danych. Zmienną możemy porównać do szuflady, w której coś chowamy, pytamy się co tam jest, zmieniamy zawartość. Także Scratch korzysta ze zmiennych, np. w kategorii **Ruch** możemy pobrać wartość aktualnych współrzędnych położenia duszka lub jego kierunek.

1. Tworzenie zmiennej

Jeśli jeszcze nie utworzyliśmy żadnej własnej zmiennej, w kategorii **Dane** będą dostępne dwa przyciski.



Drugi przycisk **Stwórz listę** służy do tworzenia zmiennych przechowujących bardziej skomplikowane dane, których nie używamy w scenariuszach zajęć opracowanych dla uczniów klas IV-VI w ramach programu „Mistrzowie Kodowania”. Na zmiennych będziemy przechowywać tylko dane liczbowe. Po kliknięciu w przycisk **Stwórz zmienną** pojawi się na ekranie okno dialogowe, jak na poniższym rysunku. Musimy określić nazwę zmiennej oraz wybrać dostępność – czy zmienna ma być dostępna dla wszystkich duszków (także dla sceny), czy tylko dla danego duszka (inne duszki nie widzą wówczas tej zmiennej, oznacza to, że mogą mieć również zmienną o takiej samej nazwie, ale będą to różne zmienne).

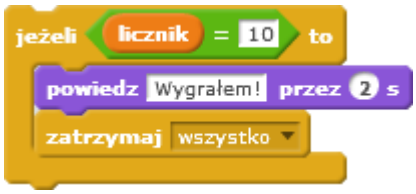


Po zatwierdzeniu przyciskiem **OK** pojawią się bloczki umożliwiające operacje na zmiennych.

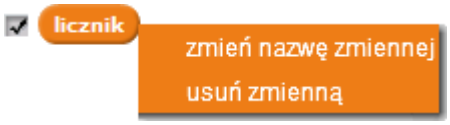
2. 

Aktualna wartość zmiennej. Można ją wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie aktualnej wartości zmiennej na scenie.

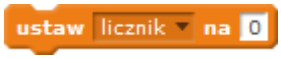
Przykład:



Po kliknięciu prawym przyciskiem myszy w klocek z nazwą zmiennej w obszarze danych z menu kontekstowego możemy wybrać jedną z dwóch opcji: zmianę nazwy zmiennej lub jej usunięcie.



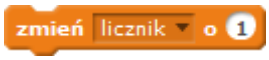
---

3. 

Przypisuje wartość zmiennej. Domyślna wartość można edytować, można też w to miejsce wstawić wyrażenie.

Przykład:

---

4. 

Zmienia wartość zmiennej o podaną wartość. W przykładzie powyżej zmienna **licznik** jest powiększana o jeden przy każdym dotknięciu duszka **mysz**.

---

5. 

Powoduje, że wartość zmiennej będzie wyświetlana na scenie. Działa analogicznie, jak edycyjne włączenie znacznika przy nazwie zmiennej. Po kliknięciu prawym przyciskiem myszy w zmienną na scenie z menu kontekstowego możemy wybrać sposób wyświetlania wartości.

„Normal readout” oznacza standardowe wyświetlanie, aktualna wartość zmiennej łącznie z nazwą, możliwość edycji wartości. „Large readout” oznacza wyświetlanie jedynie wartości (bez nazwy zmiennej) powiększoną czcionką, z możliwością edycji. Wybranie opcji suwak powoduje pojawienie się dodatkowo suwaka do zmiany wartości zmiennej. Ostatnia opcja powoduje ukrycie wyświetlania wartości na scenie. Jeśli wyświetlany jest suwak, w menu kontekstowym dostępna jest dodatkowa opcja umożliwiająca wybór minimalnej i maksymalnej wartości dla suwaka.

6. 

Powoduje, że wartość zmiennej nie będzie wyświetlana na scenie. Działa analogicznie, jak edycyjne wyłączenie znacznika przy nazwie zmiennej.

ZDARZENIA

Kategoria **Zdarzenia** zawiera klocki rozpoczynające skrypty reagujące na określone zdarzenia (np. kliknięcie duszka myszką, naciśnięcie klawisza na klawiaturze) oraz związane z obsługą komunikatów (oznacza to samodzielne generowanie zdarzeń w programie). Poniżej znajduje się opis prawie wszystkich bloczków tej kategorii.

1. 

Podstawowy bloczek. Od niego zaczyna się większość skryptów. Skrypty są uruchamiane po kliknięciu na ekranie ikony zielonej flagi – uruchomienie programu.

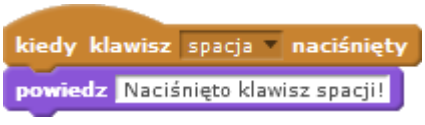
Przykład:

---

2. 

Bloczek uruchamia skrypt po naciśnięciu określonego klawisza na klawiaturze. Klawisz wybieramy z listy rozwijalnej.

Przykład:



---

3. 

Bloczek uruchamia skrypt po kliknięciu danego duszka lewym przyciskiem myszy.

Przykład:

---

4. 

Bloczek uruchamia skrypt wtedy, gdy nastąpi zmiana tła. Nazwę tła wybieramy z listy rozwijalnej. Skrypt zostanie uruchomiony wtedy, gdy to tło stanie się aktualnym (wyświetlanym na scenie).

Przykład:

---

5. 

Bloczek uruchamia skrypt wtedy, gdy nastąpi nadanie komunikatu o określonej nazwie wybranej z listy rozwijalnej. Komunikat definiujemy w bloczku **nadaj komunikat**. Nadawanie i odbieranie komunikatów umożliwia porozumiewanie się pomiędzy duszkami, a także sceną. W ten sposób jeden duszek może np. uruchomić skrypt innego duszka.

Przykład:

Skrypt kotka:

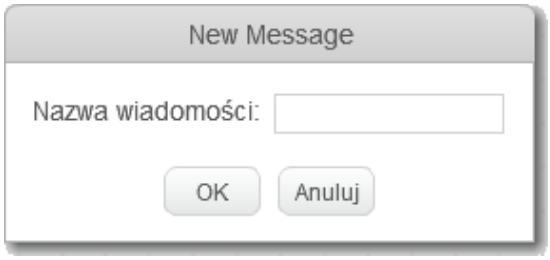
Skrypt myszki:

---

6. 

Bloczek umożliwia zdefiniowanie i nadanie komunikatu. Nazwę komunikatu możemy wybrać z listy rozwijalnej lub zdefiniować nowy komunikat przy pomocy opcji **nowy komunikat...** dostępnej po rozwinięciu listy.

Wyświetlone będzie wówczas okno dialogowe, w którym podajemy nazwę nowego komunikatu.



Po nadaniu komunikatu (wykonaniu tego bloczka) uruchamiane są automatycznie skrypty zaczynające się od bloczków **kiedy otrzymam *nazwa komunikatu***. Działanie skryptu, który nadał komunikat jest kontynuowane równolegle. Przykład: patrz opis polecenia **kiedy otrzymam**.

---

7. 

Działa analogicznie jak bloczek **nadaj *nazwa komunikatu***. (patrz opis tego polecenia). Różnica polega na tym, że działanie skryptu, w którym nadano dany komunikat wstrzymywane jest do czasu zakończenia działania wszystkich skryptów odbierających ten komunikat.

KONTROLA

Kategoria **Kontrola** zawiera bardzo ważne klocki – odpowiedniki instrukcji sterujących języków programowania (pętle, instrukcje warunkowe). Poniżej znajduje się opis wszystkich bloczków tej kategorii za wyjątkiem klonowania duszków.

1. 

Bloczek powoduje wstrzymanie działania skryptu na określony czas. Domyślną wartość możemy edytować.

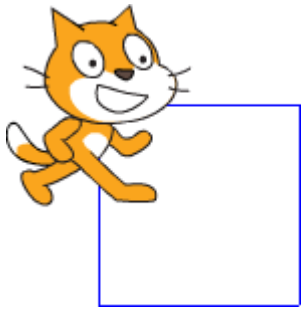
Przykład:

---

2. 

Bloczek powoduje powtarzanie zestawu instrukcji określoną liczbę razy. Domyślną liczbę powtórzeń możemy edytować.

Przykład:

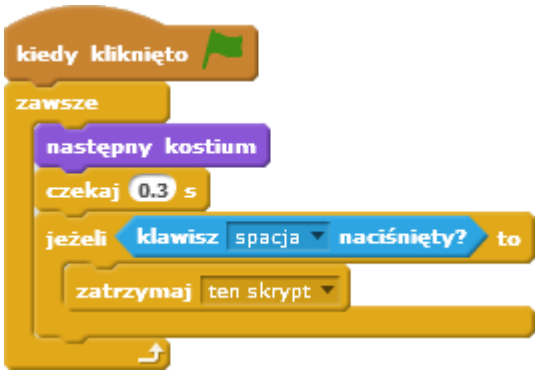


---

3. 

Bloczek powoduje powtarzanie zestawu instrukcji w nieskończoność, do naciśnięcia przycisku przerwania – czerwonego koła. Jeśli skrypt z bloczkiem **zawsze** ma się zatrzymać sam, należy w jego wnętrzu użyć instrukcji warunkowej i bloczka **zatrzymaj**.

Przykład:

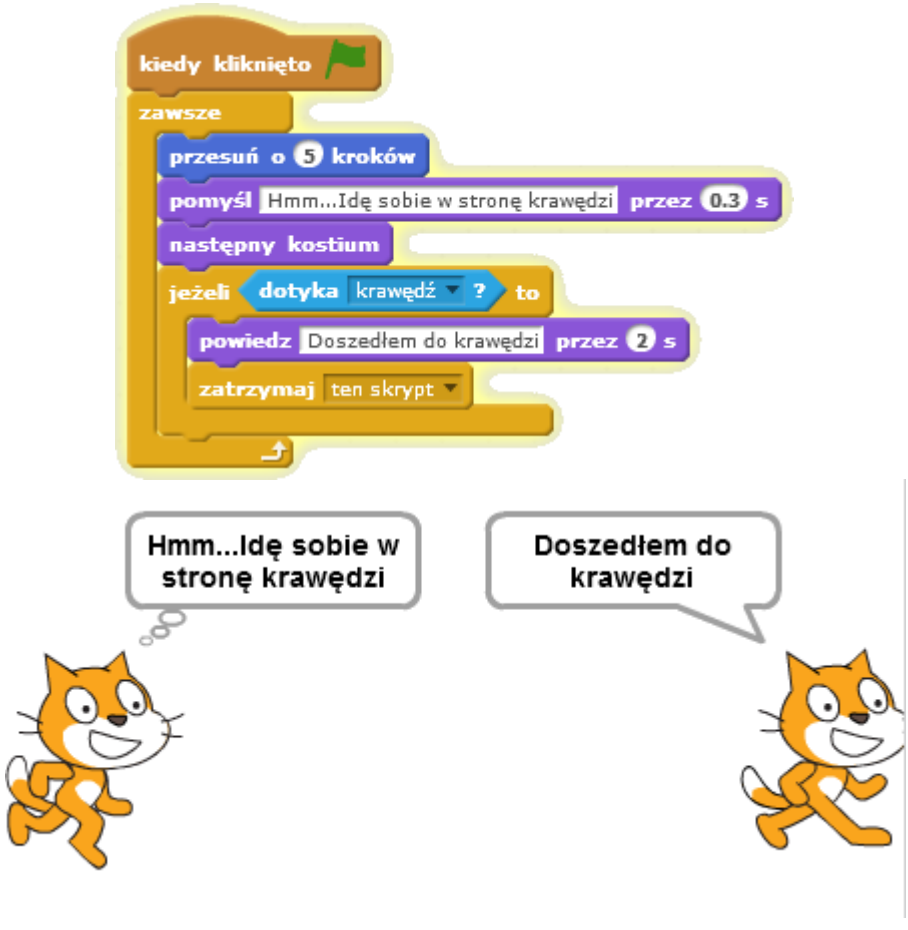


---

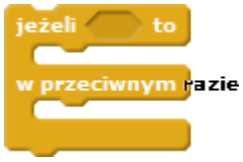
4. 

Bloczek instrukcji warunkowej, wykonany zostanie raz zestaw instrukcji, jeżeli warunek logiczny wstawiony po słowie **jeżeli** jest prawdziwy. Warunki budujemy z bloczków kategorii **Czujniki** i **Wyrażenia**.

Przykład:

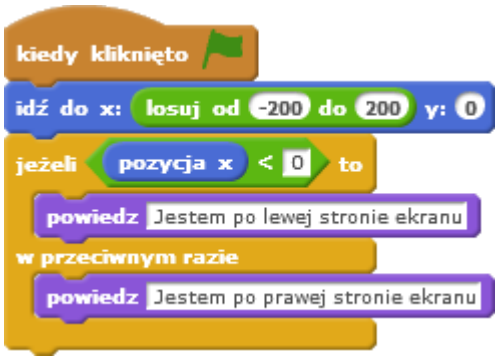


5.



Bloczek instrukcji warunkowej, wykonany zostanie zestaw instrukcji pomiędzy słowami **to** i **w przeciwnym razie**, jeżeli warunek logiczny wstawiony po słowie **jeżeli** jest prawdziwy. W przeciwnym przypadku zostanie wykonany zestaw instrukcji po słowie **w przeciwnym razie**. Warunki budujemy z bloczków kategorii **Czujniki** i **Wyrażenia**.

Przykład:

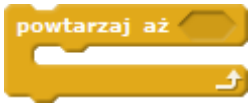


6.

Wstrzymuje działanie skryptu do momentu spełnienia warunku logicznego. Warunki budujemy z bloczków kategorii **Czujniki** i **Wyrażenia**.

Przykład:

7.



Bloczek powoduje powtarzanie zestawu instrukcji aż do spełnienia warunku logicznego. Warunki budujemy z bloczków kategorii **Czujniki** i **Wyrażenia**.

Przykład:

---

8. 

Powoduje zatrzymanie wykonywania skryptu lub skryptów. Z listy rozwijalnej można wybrać: zatrzymanie skryptu, w którym umieszczono bloczek (**ten skrypt**), wszystkich skryptów wszystkich duszków i sceny (**wszystko**) oraz innych skryptów danego duszka za wyjątkiem skryptu, w którym został uruchomiony ten bloczek (**inne skrypty duszka**). Przykłady: patrz opis bloczków **zawsze** i **jeżeli**.

CZUJNIKI

Kategoria **Czujniki** zawiera bloczki związane z rozpoznawaniem różnych sytuacji zachodzących na scenie, dotyczących m.in. duszków i myszki, bloczki umożliwiające pobieranie danych z klawiatury, bloczki związane z zegarem, datą oraz kamerą. Poniżej znajduje się opis większości bloczków tej kategorii za wyjątkiem obsługi kamery oraz bloczków daty.

1. 

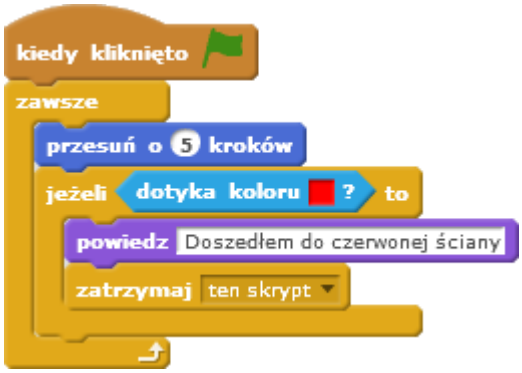
Sprawdza, czy duszek dotyka elementu wybranego z listy rozwijalnej. Może to być dowolny inny duszek, kursor myszy lub krawędź sceny. Bloczek ten jest wykorzystywany najczęściej w połączeniu z jednym z bloczków **jeżeli**, **powtarzaj aż**, **czekaj aż**.

Przykład:

2. 

Sprawdza, czy duszek dotyka określonego koloru na scenie. Aby wybrać kolor, należy kliknąć myszką w kwadracik z kolorem w bloczku, następnie wybrać kolor kursorem myszy z dostępnych na ekranie. Bloczek ten jest wykorzystywany najczęściej w połączeniu z jednym z bloczków **jeżeli**, **powtarzaj aż**, **czekaj aż**.

Przykład:



---

3. czy kolor dotyka ?

Sprawdza, czy wybrany kolor postaci duszka dotyka określonego koloru na scenie. Aby wybrać kolor, należy kliknąć myszką w kwadracik z kolorem w bloczku, następnie wybrać kolor kursorem myszy z dostępnych na ekranie. Bloczek ten jest wykorzystywany najczęściej w połączeniu z jednym z bloczkami **jeżeli**, **powtarzaj aż**, **czekaj aż**.

Przykład:

---

4. odległość do

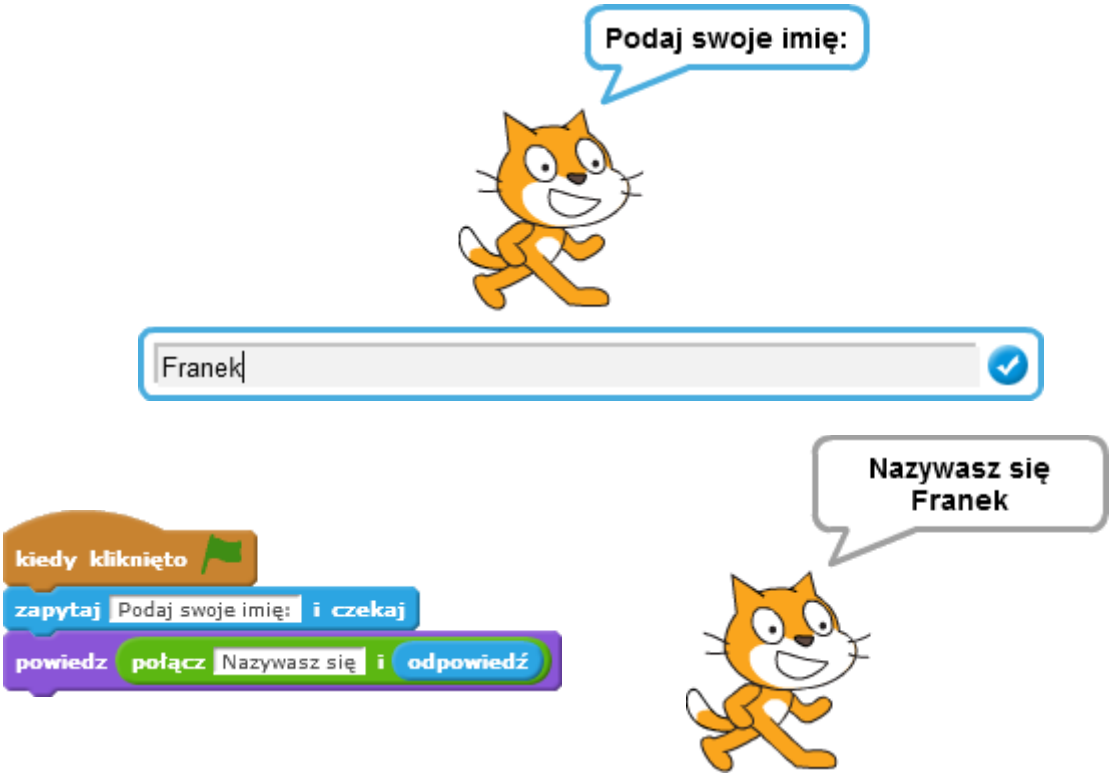
Aktualna odległość danego duszka od elementu wybranego z listy rozwijalnej – kursora myszy lub innego duszka. Można ją wykorzystać jako argument w wyrażeniach. Odległość jest liczona od środka duszka – można go ustawić edytując kostium.

Przykład:

5. zapytaj What's your name? i czekaj

Wyświetla w dymku tekst wpisany w pole bloczka oraz okienko na scenie z polem edycyjnym. Czeka na wprowadzenie danych w polu edycyjnym i zatwierdzenie klawiszem Enter lub przyciskiem w polu edycyjnym. Wprowadzona z klawiatury informacja dostępna jest w skryptach za pomocą bloczka **odpowiedź**.

Przykład:



6. odpowiedź

Podana przez użytkownika odpowiedź, jako wynik działania klocka **Zapytaj i czekaj**. Można ją wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie podanej odpowiedzi na scenie. Przykład: patrz opis bloczka **Zapytaj i czekaj**.

7. 

Sprawdza, czy wybrany z listy rozwijalnej klawisz został naciśnięty. Bloczek ten jest wykorzystywany najczęściej w połączeniu z jednym z bloczków **jeżeli**, **powtarzaj aż**, **czekaj aż**.

Przykład:

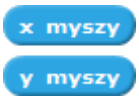
---

8. 

Sprawdza, czy został wciśnięty lewy klawisz myszy. Bloczek ten jest wykorzystywany najczęściej w połączeniu z jednym z bloczków **jeżeli**, **powtarzaj aż**, **czekaj aż**.

Przykład:

---

9. 

Aktualna wartość x-owej oraz y-owej współrzędnej kursora myszy. Można je wykorzystać jako argument w wyrażeniach. Współrzędne te są także wyświetlane pod prawym dolnym rogiem sceny.

Przykład:

---

10. czasomierz

Aktualna czas liczony w sekundach od momentu ostatniego kliknięcia zielonej flagi lub wykonania bloczka **kasuj zegar**. Można go wykorzystać jako argument w wyrażeniach. Włączenie znacznika po lewej stronie powoduje wyświetlanie czasu na scenie.

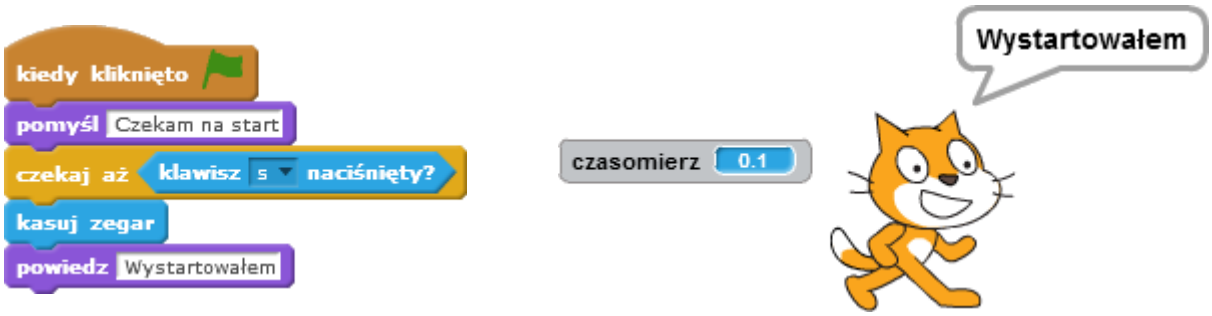
Przykład:

---

11. kasuj zegar

Zeruje odliczanie czasu. Wartości oddawane przez bloczek czasomierz będą ponownie liczone od zera.

Przykład:

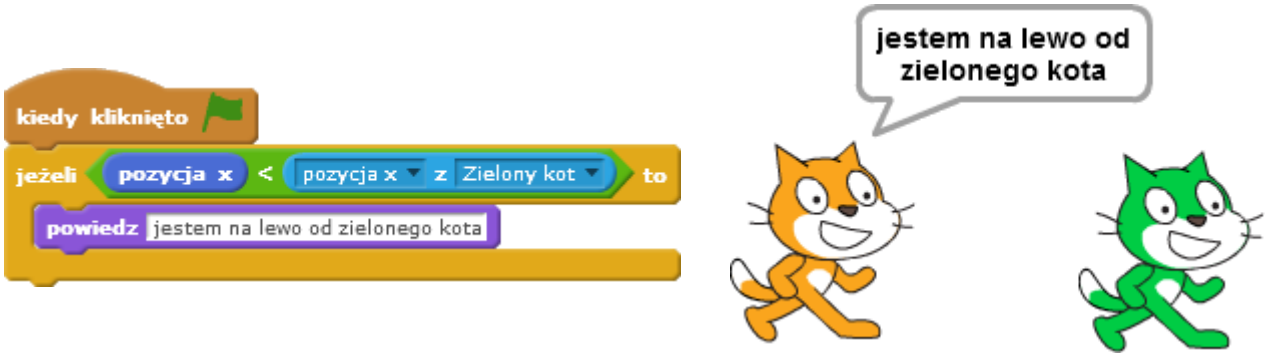


---

12. pozycja x z Sprite1

Aktualna wartość wybrana z pierwszej listy rozwijalnej, dotycząca duszka wybranego z drugiej listy lub sceny. Można wybrać x-ową lub y-ową współrzędną, kierunek, numer kostiumu lub nazwę kostiumu, rozmiar. Dla sceny do wyboru są numer lub nazwa tła. Oddawaną wartość można wykorzystać w wyrażeniach.

Przykład:



WYRAŻENIA

W kategorii **Wyrażenia** zgrupowane są bloczki podstawowych działań arytmetycznych, operacji logicznych oraz różnych funkcji (zarówno arytmetycznych jak i na tekstach). Wszystkie bloczki tej kategorii są wykorzystywane jako argumenty do innych bloczków. Poniżej znajduje się opis większości bloczków tej kategorii.

1. 

Bloczki czterech podstawowych operacji arytmetycznych (dodawanie, odejmowanie, mnożenie, dzielenie). W polach edycyjnych należy podać argumenty (mogą to być ręcznie wpisane liczby, wartość zmiennej lub wynik innego działania, a więc bloczki można zagnieżdżać). Zbudowane wyrażenie arytmetyczne należy wykorzystać jako argument innego bloczka, np. określenia wartości zmiennej, czy wyrażenia logicznego.

Przykład:

2. 

Losuje liczbę z podanego zakresu. W polach edycyjnych należy podać dolny i górny zakres losowania. Bloczek należy wykorzystać jako argument innego bloczka.

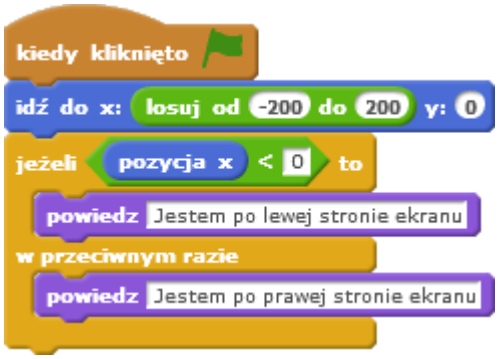
Przykład:

---

3. 

Bloczki porównania (mniejsze, równe, większe). W polach edycyjnych należy podać porównywane wartości. Można wpisać konkretne wartości w pola edycyjne, najczęściej jednym z argumentów jest wartość zmiennej lub wartość odczytywana przy pomocy innego bloczka (np. współrzędna położenia duszka na scenie). Bloczek jest wykorzystywany najczęściej w instrukcjach **jeżeli, powtarzaj aż, czekaj aż**.

Przykład:



---

4. 

Bloczki służące do budowania złożonych wyrażeń logicznych (koniunkcja – obydwa warunki są prawdziwe, alternatywa – jeden z warunków jest prawdziwy, negacja – zaprzeczenie). W pola argumentów należy wstawić pasujące bloczki zwracające wartości logiczne (prawda lub fałsz). Bloczki są wykorzystywane najczęściej w instrukcjach **jeżeli, powtarzaj aż, czekaj aż**.

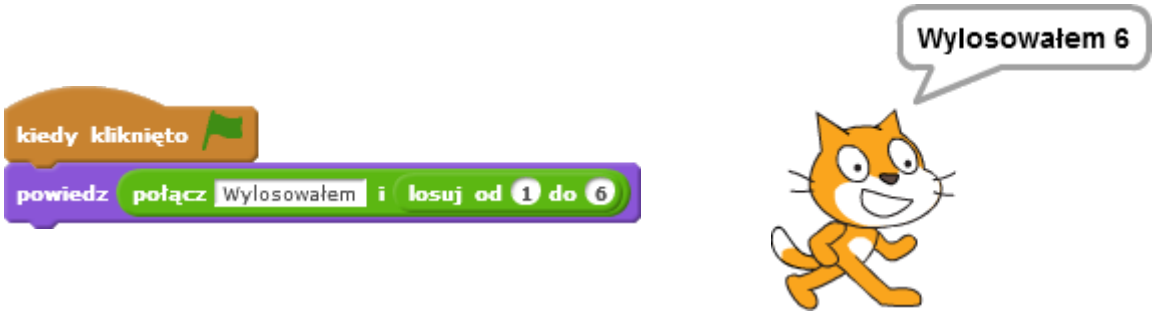
Przykład:

---

5. połącz hello i world

Bloczek służący do połączenia dwóch napisów w jeden. Najczęściej jeden z argumentów jest wpisanym tekstem, a drugi wartością zwracaną przez inny bloczek. Bloczek jest wykorzystywany często jako argument w bloczkach **powiedz** i **pomyśl**.

Przykład:

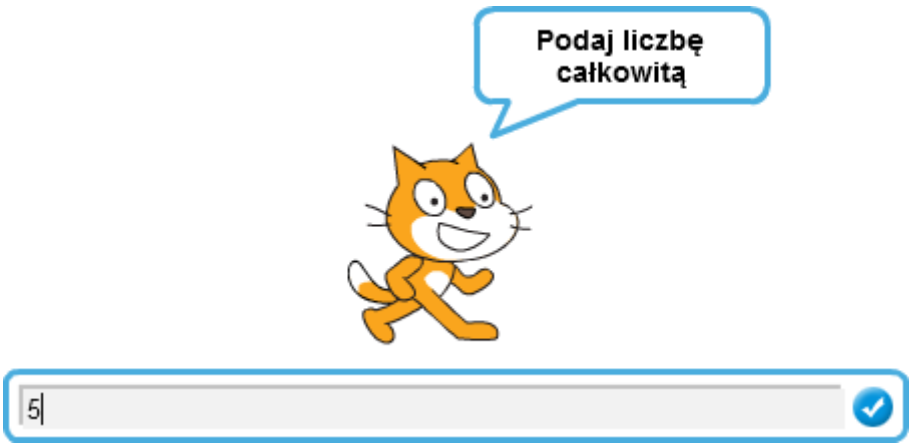


---

6. mod

Wartością jest reszta z dzielenia pierwszego argumentu przez drugi. Bloczek jest wykorzystywany jako argument w wyrażeniach arytmetycznych i logicznych.

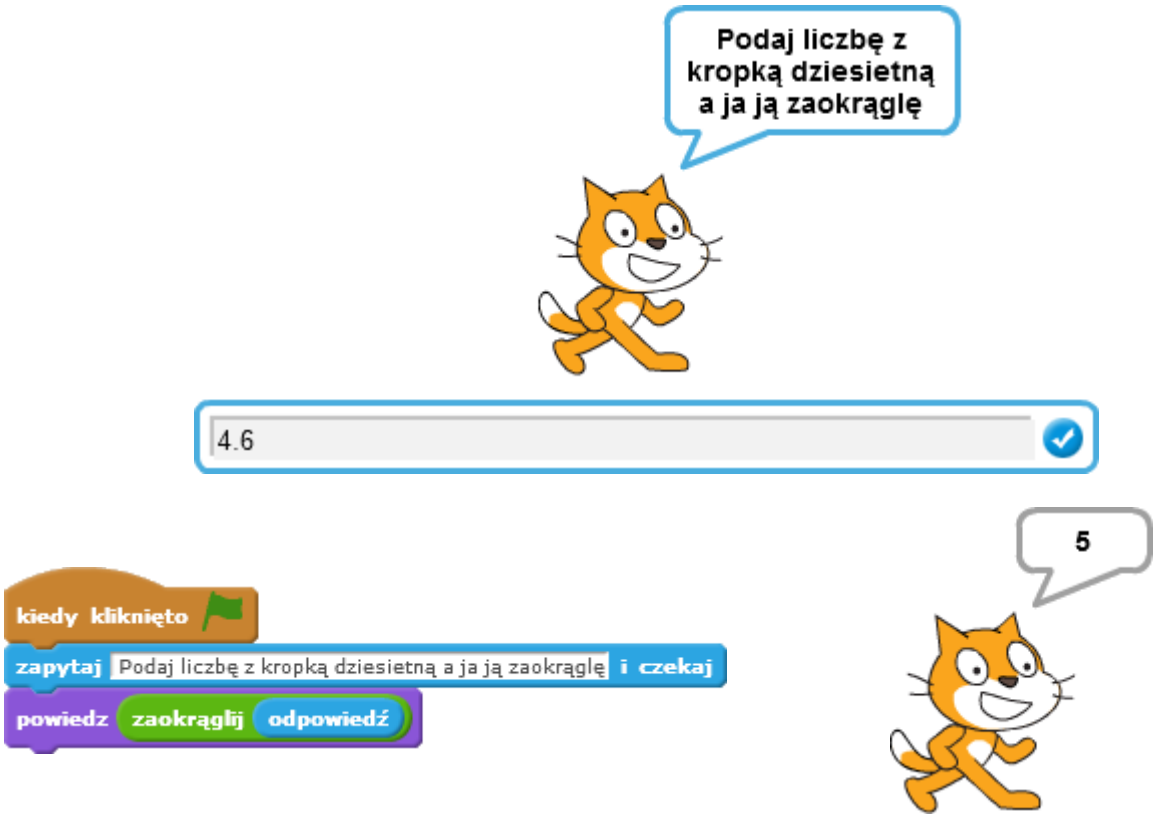
Przykład:



7. zaokrąglij

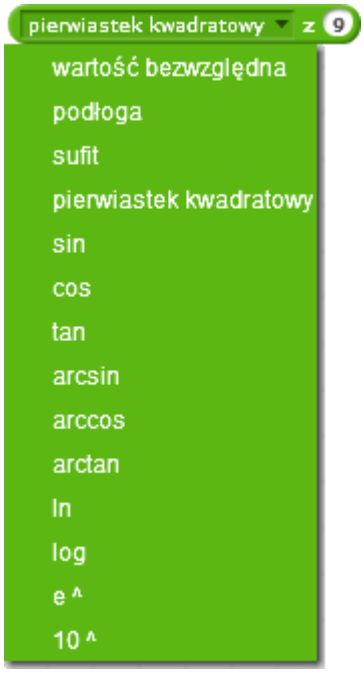
Wartością jest zaokrąglony argument do najbliższej liczby całkowitej – zgodnie z regułami zaokrąglania, tzn. 0.5 i więcej jest zaokrąglana w górę, w przeciwnym przypadku w dół. Bloczek jest wykorzystywany jako argument w innych bloczkach.

Przykład:



8. pierwiastek kwadratowy z 9

Wartością jest wynik funkcji wybranej z listy rozwijalnej dla argumentu podanego w polu edycyjnym. Z listy rozwijalnej można wybrać następujące funkcje:



Bloczek jest wykorzystywany jako argument w innych bloczkach. Wyjaśnienia mogą wymagać funkcje o nazwach **podłoga** (zaokrąglenie zawsze w dół) oraz **sufit** (zaokrąglenie zawsze w górę).

Przykład:

BLOKI SCENY

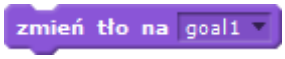
Scena, podobnie jak duszki, może mieć własne skrypty. Aby tworzyć skrypty dla sceny trzeba ją wybrać jako aktywny obiekt w prawym dolnym roku ekranu – obszarze zarządzania duszkami i sceną. Zmieni się wówczas repertuar dostępnych bloków – wiele z nich dotyczy wyłącznie duszków i nie można ich wykorzystywać w skryptach sceny. Poniżej opisano występujące różnice.

1. Kategoria **Ruch**

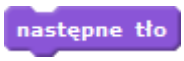
   Nie ma w tej kategorii żadnych bloczków, sceny nie można przesuwać ani obracać.

2. Kategoria **Wygląd**

   Najwięcej zmian jest w tej kategorii. Scena może posiadać wiele teł, podobnie jak duszek kostiumów. Dostępne są bloczki związane ze zmianą tła oraz informacje o wybranym tle. Najważniejsze z nich to:

   Zmiana tła na wybrane z listy rozwijalnej. Tła sceny przypisujemy w zakładce tła. Możemy załadować tło z galerii, narysować, wybrać z pliku na dysku bądź sfotografować przy pomocy kamery. W przypadku importowania jako tło plików graficznych należy pamiętać o wymiarach sceny – 480 na 360 punktów.

   Zmiana tła na następne. Następnym tłem po ostatnim jest ponownie pierwsze.

   Nazwa aktualnie wyświetlanego tła.

   Numer aktualnie wyświetlanego tła.

3. Kategoria **Dźwięk**

   Bloczki są takie same, ale w przypadku odtwarzania plików dźwiękowych odnoszą się do dźwięków przypisanych scenie w zakładce **Dźwięki**.

4. Kategoria **Pisak**

   W tej kategorii dla sceny jest tylko jeden bloczek – **wyczyść**.

5. Kategoria **Dane**

   Tak samo jak dla duszków. Jedyna zmiana, to taka, że można tu tworzyć zmienne dostępne tylko dla wszystkich duszków.

6. Kategoria **Zdarzenia**

   Takie same bloczki jak dla duszków.

7. Kategoria **Kontrola**

   Takie same bloczki jak dla duszków.

8. Kategoria **Czujniki**

   Takie same bloczki jak dla duszków. Nie występują bloczki dotyczące oddziaływania duszków z innymi obiektami i kolorami (**dotyka**, **dotyka koloru**, **czy kolor dotyka koloru**, **odległość od**)

9. Kategoria **Wyrażenia**

   Takie same bloczki jak dla duszków.