turtle— Želví grafika

Pohyb želvy

turtle.forward(distance)
turtle.fd(distance)

vzdálenost – číslo (celé číslo nebo plovoucí)

Posuňte želvu dopředu o zadanou *vzdálenost* ve směru a želva má v čele.

turtle.back(distance)
turtle.bk(distance)
turtle.backward(distance)

vzdálenost – číslo

Posuňte želvu vzad o *vzdálenost* , proti směru a želva má v čele. Neměňte směr želvy.

turtle.right(angle)
turtle.rt(angle)

úhel – číslo (celé číslo nebo plovoucí)

Otočte želvu doprava o *jednotky úhlu* . (Jednotky jsou ve výchozím nastavení stupně, ale lze nastavit pomocí degrees()a radians()funkce.) Úhel orientace závisí na režimu želvy, viz mode().

turtle.left(angle)
turtle.lt(angle)

úhel – číslo (celé číslo nebo plovoucí)

Otočte želvu doleva o *jednotky úhlu* . (Jednotky jsou ve výchozím nastavení stupně, ale lze nastavit pomocí degrees()a radians()funkce.) Úhel orientace závisí na režimu želvy, viz mode().

turtle.goto(x, y=None)
turtle.setpos(x, y=None)
turtle.setposition(x, y=None)

x – číslo nebo dvojice/vektor čísel

y – číslo nebo None

Přesuňte želvu do absolutní polohy. Pokud je pero dole, nakreslete čáru. Dělat nemění orientaci želvy.

turtle.setx(x)

x – číslo (celé číslo nebo plovoucí)

Nastavte první souřadnici želvy na *x* , ponechte druhou souřadnici beze změny.

turtle.sety(y)

y – číslo (celé číslo nebo plovoucí)

Nastavte druhou souřadnici želvy na *y* , první souřadnici ponechte beze změny.

turtle.setheading(to_angle)
turtle.seth(to_angle)

to_angle – číslo (celé číslo nebo plovoucí)

Nastavte orientaci želvy na *to_angle* . Zde jsou některé běžné směry ve stupních:

standardní režim	režim loga
0 - východ	0 - sever
90 - sever	90 - východ
180 - západ	180 - jih
270 - jih	270 - západ

turtle.home()

Přesuňte želvu na počátek – souřadnice (0,0) – a nastavte její směr na jeho počáteční orientace (která závisí na režimu, viz mode()).

turtle.circle(radius, extent=None, steps=None)

poloměr – číslo
rozsah – číslo (příp None)
kroky – celé číslo (příp None)
Nakreslete kružnici s daným *poloměrem*

turtle.stamp()

Vyrazte kopii tvaru želvy na plátno u aktuální želvy pozice. Vraťte razítko_id pro toto razítko, které lze použít k odstranění to zavoláním clearstamp(stamp_id).

turtle.clearstamp(stampid)

stampid – celé číslo, musí být návratovou hodnotou předchozí stamp()volání
Smazat razítko s daným *razítkem* .

turtle.clearstamps(n=None)

n – celé číslo (nebo None)
Smazat všechna nebo prvních/posledních *n* razítek želvy. Pokud *n* je None, smazat všechna razítka, je-li *n* > 0, odstraní prvních *n* razítek, jinak pokud *n* < 0 odstraní posledních *n* známek.

turtle.undo()

Vraťte zpět (opakovaně) poslední akci želvy. Počet dostupných akce zpět je určena velikostí undobuffer.

turtle.speed(speed=None)

rychlost – celé číslo v rozsahu 0..10 nebo speedstring (viz níže)
Nastavte rychlost želvy na celočíselnou hodnotu v rozsahu 0..10. Jestli ne je dán argument, vrátí aktuální rychlost.
Pokud je na vstupu číslo větší než 10 nebo menší než 0,5, nastaví se rychlost na 0. Rychlostní řetězce jsou mapovány na hodnoty rychlosti takto:
"nejrychlejší": 0 , "rychle": 10 , "normální": 6 , "pomalé": 3 , "nejpomalejší": 1
Rychlosti od 1 do 10 vynucují stále rychlejší animaci kreslení čar a otáčení želvy.
Pozor: *rychlost* = 0 znamená, že *žádná* animace nebere místo. vpřed/vzad skočí želva a podobně vlevo/vpravo skočí želva se okamžitě otočí.

Řekněte želví stav

turtle.position()

turtle.pos()

Vraťte aktuální polohu želvy (x,y) (jako a Vec2Dvektor).

turtle.towards(x, y=None)

x – číslo nebo dvojice/vektor čísel nebo instance želvy
y – číslo, pokud *x* je číslo, jinak None
Vraťte úhel mezi úsečkou z pozice želvy do určené pozice pomocí (x,y), vektoru nebo jiné želvy. To závisí na startu želvy orientace, která závisí na režimu - „standard“/“svět“ nebo „logo“.

turtle.xcor()

Vraťte želvu souřadnici x.

turtle.ycor()

Vraťte souřadnici y želvy.

turtle.heading()

Vraťte aktuální kurz želvy (hodnota závisí na režimu želvy, viz mode()).

turtle.distance(x, y=None)

x – číslo nebo dvojice/vektor čísel nebo instance želvy
y – číslo, pokud *x* je číslo, jinak None
Vraťte vzdálenost od želvy k (x,y), danému vektoru nebo danému jiná želva, v jednotkách kroku želvy.

..

Nastavení pro měření

turtle.degrees(fullcircle=360.0)

celý kruh – číslo

Nastavte jednotky měření úhlu, tj. nastavte počet „stupňů“ pro celý kruh. Výchozí hodnota je 360 stupňů.

turtle.radians()

Nastavte jednotky měření úhlu na radiány. Ekvivalentní degrees(2*math.pi).

Ovládání perem

turtle.pendown()

turtle.pd()

turtle.down()

Táhněte pero dolů – kreslení při pohybu.

turtle.penup()

turtle.pu()

turtle.up()

Vytáhněte pero nahoru – žádné kreslení při pohybu.

turtle.pensize(width=None)

turtle.width(width=None)

šířka – kladné číslo

Nastavte tloušťku čáry na *šířku* nebo ji vraťte. Pokud je nastaven režim změny velikosti na „auto“ a tvar želvy je mnohoúhelník, tento mnohoúhelník je nakreslen stejnou čarou tloušťka. Pokud není zadán žádný argument, vrátí se aktuální velikost penze.

turtle.pen(pen=None, **pendict)

pero – slovník s některými nebo všemi níže uvedenými klávesami

pendict – jeden nebo více argumentů klíčových slov s níže uvedenými klíči jako klíčovými slovy

Vraťte nebo nastavte atributy pera ve „slovníku pera“ následujícím způsobem páry klíč/hodnota: „zobrazeno“: Pravda/nepravda , „penddown“: Pravda/nepravda , „barva pera“: barevný řetězec nebo barevný n-tice , „fillcolor“: barevný řetězec nebo color-tuple , „pensize“: kladné číslo , „speed“: číslo v rozsahu 0..10 , „resizemode“: „auto“ nebo „user“ nebo „noresize“ , „stretchfactor“: (kladné číslo, kladné číslo) , „obrys“: kladné číslo , „naklonit“: číslo

Tento slovník lze použít jako argument pro následující volání pen()obnovit původní stav pera. Navíc jeden nebo více těchto atributů lze poskytnout jako argumenty klíčových slov. To lze použít k nastavení několika per atributy v jednom příkazu.

turtle.isdown()

Vrátit se Truepokud je pero dole, Falsejestli je nahoře.

Ovládání barev

turtle.pencolor(*args)

Vraťte nebo nastavte barvu pera.

turtle.fillcolor(*args)

Vraťte nebo nastavte barvu výplně.

turtle.color(*args)

Vraťte nebo nastavte barvu pera a barvu výplně.

Je povoleno několik vstupních formátů.

Pokud je tvar želvy mnohoúhelník, nakreslí se obrys a vnitřek tohoto mnohoúhelníku s nově nastavenými barvami.

Výplně

turtle.filling()

Návrat fillstate (Truepokud se plní, Falsejiný).

turtle.begin_fill()

Vyvolá se těsně před nakreslením tvaru, který má být vyplněn.

turtle.end_fill()

Vyplňte tvar nakreslený po posledním volání begin_fill().

Zda se překrývají oblasti pro samo se protínající polygony nebo je vyplněno více tvarů závisí na grafice operačního systému, typ překrytí a počet překrytí. Například hvězda Turtle výše může být buď celá žlutá, nebo mít některé bílé oblasti.

Více ovládání kreslení

turtle.reset()

Odstraňte kresby želvy z obrazovky, znovu želvu vycentrujte a nastavte proměnné na výchozí hodnoty.

turtle.clear()

Odstraňte kresby želvy z obrazovky. Nehýbejte želvou. Stát a pozice želvy ani kresby ostatních želv nejsou ovlivněny.

turtle.write(arg, move=False, align='left', font=('Arial', 8, 'normal'))

arg – objekt, který má být zapsán na TurtleScreen

pohyb – pravda/nepravda

zarovnat – jeden z řetězců „vlevo“, „na střed“ nebo vpravo“

písmo – trojice (název písma, velikost písma, typ písma)

Napište text - řetězcové vyjádření *arg* - u aktuální želvy pozice podle *zarovnání* („vlevo“, „uprostřed“ nebo „vpravo“) as daným písmo. Pokud *je pohyb* pravdivý, pero se přesune do pravého dolního rohu text. Ve výchozím nastavení *přesun* je False.

Stav želvy

turtle.hideturtle()

turtle.ht()

Udělejte želvu neviditelnou. Je to dobrý nápad udělat to, když jste v uprostřed nějaké složité kresby, protože skrytí želvy urychluje kreslení pozorovatelně.

turtle.showturtle()

turtle.st()

Zviditelnit želvu.

turtle.isvisible()

Vrátit se Truepokud je zobrazena želva, Falsepokud je to skryté.

turtle.shape(name=None)

name – řetězec, který je platným názvem tvaru

Nastavte tvar želvy na tvar s křestním *jménem* nebo, pokud jméno není zadáno, vraťte se název aktuálního tvaru. Tvar s *názvem* musí existovat na TurtleScreen tvarový slovník.

turtle.resizemode(rmode=None)

rmode – jeden z řetězců „auto“, „user“, „noresize“

Nastavte resizemode na jednu z hodnot: „auto“, „user“, „noresize“. Pokud *rmode* není zadáno, vrátí aktuální resizemode. Různé režimy změny velikosti mají následující efekty:

„auto“: přizpůsobí vzhled želvy odpovídající hodnotě pensize.

„uživatel“: přizpůsobí vzhled želvy podle hodnot stretchfactor a outlinewidth (outline), které jsou nastaveny pomocí shapesize().

„noresize“: nedochází k žádné úpravě vzhledu želvy.

resizemode("user")je volán podle shapesize()při použití s argumenty.

turtle.shapesize(stretch_wid=None, stretch_len=None, outline=None)
turtle.turtlesize(stretch_wid=None, stretch_len=None, outline=None)

stretch_wid – kladné číslo
stretch_len – kladné číslo
osnova – kladné číslo
Vraťte nebo nastavte atributy pera x/y-stretchfactors a/nebo obrys. Soubor resizemode na „user“. Pokud a pouze tehdy, je-li resizemode nastaven na „user“, želva se zobrazí natažený podle jeho stretchfactors: *stretch_wid* is stretchfactor kolmý k jeho orientaci, *stretch_len* je stretchfactor ve směru jeho orientace, *obrys* určuje šířku obrysu tvarů.

turtle.shearfactor(shear=None)

nůžky – číslo (volitelné)
Nastavte nebo vraťte aktuální smykový faktor. Tvar želvy ostříhejte podle daný smykový faktor smyk, který je tečnou úhlu smyku. Neměňte . směr želvy (směr pohybu) Pokud není zadán smyk: vraťte aktuální smykový faktor, tj tečna úhlu smyku, o kterou přímky rovnoběžné s želvy jsou oříznuty.

turtle.tilt(angle)

úhel – číslo
Otočte tvar želvy o *úhel* z jeho aktuálního úhlu naklonění, ale *ne* změnit směr pohybu želvy (směr pohybu).

turtle.settiltangle(angle)

úhel – číslo
Otočte tvar želvy tak, aby ukazoval ve směru určeném *úhlem* , bez ohledu na jeho aktuální úhel sklonu. *Neměňte* směr želvy (směr pohybu).

turtle.tiltangle(angle=None)

úhel – číslo (volitelné)
Nastavte nebo vraťte aktuální úhel náklonu. Pokud je zadán úhel, otočte tvar želvy, aby ukazoval ve směru určeném úhlem, bez ohledu na jeho aktuální úhel sklonu. neměňte _ Želvě kurz (směr pohybu). Pokud není zadán úhel: vraťte aktuální úhel náklonu, tj. úhel mezi orientací tvaru želvy a směřováním želva (směr jejího pohybu).

turtle.shapetransform(t11=None, t12=None, t21=None, t22=None)

t11 – číslo (volitelné)
t12 – číslo (volitelné)
t21 – číslo (volitelné)
t12 – číslo (volitelné)
Nastavte nebo vraťte aktuální transformační matici tvaru želvy.
Pokud není zadán žádný z prvků matice, vraťte transformaci matice jako n-tice 4 prvků. Jinak nastavte dané prvky a transformujte tvar želvy podle matice sestávající z prvního řádku t11, t12 a druhá řada t21, t22. Nesmí být determinant t11 * t22 - t12 * t21 nula, jinak se objeví chyba. Upravte stretchfactor, shearfactor a tiltangle podle toho daná matrice.

turtle.get_shapepoly()

Vrátí polygon aktuálního tvaru jako n-tici dvojic souřadnic. Tento lze použít k definování nového tvaru nebo součástí složeného tvaru.

Použití událostí

turtle.onclick(fun, btn=1, add=None)

fun – funkce se dvěma argumenty, která bude volána s souřadnice klepnutého bodu na plátně
btn – číslo tlačítka myši, výchozí 1 (levé tlačítko myši)
přidat – Truenebo False– pokud True, bude nová vazba přidáno, jinak nahradí dřívější vazbu
Spojte *zábavu* s událostmi kliknutí myší na této želvě. Pokud *zábava* je None, existující vazby jsou odstraněny. Příklad pro anonymní želvu, tj procedurální způsob:

turtle.onrelease(fun, btn=1, add=None)

fun – funkce se dvěma argumenty, která bude volána s souřadnice klepnutého bodu na plátně

btn – číslo tlačítka myši, výchozí 1 (levé tlačítko myši)

přidat – Truenebo False– pokud True, bude nová vazba přidáno, jinak nahradí dřívější vazbu

Spojte *zábavu* s událostmi stisku tlačítka myši na této želvě. Pokud *zábava* je None, stávající vazby jsou odstraněny.

turtle.ondrag(fun, btn=1, add=None)

fun – funkce se dvěma argumenty, která bude volána s souřadnice klepnutého bodu na plátně

btn – číslo tlačítka myši, výchozí 1 (levé tlačítko myši)

přidat – Truenebo False– pokud True, bude nová vazba přidáno, jinak nahradí dřívější vazbu

Spojte *zábavu* s událostmi pohybu myší na této želvě. Pokud *zábava* je None, existující vazby jsou odstraněny.

Poznámka: Každé sekvenci pohybů myší na želvě předchází a událost kliknutí myší na tuto želvu.

Speciální želví metody

turtle.begin_poly()

Začněte zaznamenávat vrcholy mnohoúhelníku. Aktuální pozice želvy je první vrchol mnohoúhelníku.

turtle.end_poly()

Zastavit záznam vrcholů mnohoúhelníku. Aktuální pozice želvy je poslední vrchol mnohoúhelníku. To bude spojeno s prvním vrcholem.

turtle.get_poly()

Vraťte poslední zaznamenaný polygon.

turtle.clone()

Vytvořte a vraťte klon želvy se stejnou pozicí, směrem a vlastnosti želvy.

turtle.getturtle()

turtle.getpen()

Vraťte samotný objekt Turtle. Pouze rozumné použití: jako funkce k vrátit „anonymní želvu“:

turtle.getscreen()

Vraťte TurtleScreenobjekt, na který želva kreslí. Pro tento objekt lze poté volat metody TurtleScreen.

turtle.setundobuffer(size)

velikost – celé číslo nebo None

Nastavte nebo zakažte undobuffer. Je-li *velikost* celé číslo, prázdný undobuffer of daná velikost je nainstalována. *velikost* udává maximální počet akcí želvy které lze zrušit tím undo()metoda/funkce. Pokud *velikost* je None, undobuffer je zakázán.

turtle.undobufferentries()

Vrátí počet záznamů v undobufferu.

Metody TurtleScreen/Screen a odpovídající funkce

Většina příkladů v této části odkazuje na nazvanou instanci TurtleScreen screen.

Ovládání oken

turtle.bgcolor(*args)

args – barevný řetězec nebo tři čísla v rozsahu 0..colormode nebo a 3-násobek takových čísel

Nastavte nebo vraťte barvu pozadí TurtleScreen.

turtle.bgpic(picname=None)

picname – řetězec, název souboru gif popř "nopic"nebo None

Nastavit obrázek na pozadí nebo vrátit název aktuálního obrázku na pozadí.

turtle.clear()

Tato metoda TurtleScreen je k dispozici jako globální funkce pouze pod název clearscreen. Globální funkce clearje jiný odvozené z Turtle metody clear.

turtle.clearscreen()

Odstraňte všechny kresby a všechny želvy z obrazovky TurtleScreen. Resetujte nyní prázdný TurtleScreen do původního stavu: bílé pozadí, žádné pozadí obrázek, žádné vazby událostí a sledování.

turtle.reset()

Tato metoda TurtleScreen je k dispozici jako globální funkce pouze pod název resetscreen. Globální funkce resetje další odvozené z Turtle metody reset.

turtle.resetscreen()

Obnovte všechny želvy na obrazovce do výchozího stavu.

turtle.screensize(canvwidth=None, canvheight=None, bg=None)

canvwidth – kladné celé číslo, nová šířka plátna v pixelech
canvheight – kladné celé číslo, nová výška plátna v pixelech
bg – colorstring nebo color-tuple, nová barva pozadí
Pokud nejsou zadány žádné argumenty, vrátí aktuální (canvaswidth, canvasheight). Jiný změnit velikost plátna, na které želvy kreslí. Neměňte kresbu okno. Chcete-li pozorovat skryté části plátna, použijte posuvníky. S tím lze zviditelnit ty části výkresu, které byly mimo plátno předtím.

turtle.setworldcoordinates(llx, lly, urx, ury)

llx – číslo, x-ová souřadnice levého dolního rohu plátna
lly – číslo, y-ová souřadnice levého dolního rohu plátna
urx – číslo, x-ová souřadnice pravého horního rohu plátna
ury – číslo, souřadnice y pravého horního rohu plátna
Nastavte uživatelsky definovaný souřadnicový systém a přepněte do režimu „svět“, pokud nutné. To provádí a screen.reset(). Pokud již režim „svět“ je aktivní, všechny výkresy se překreslí podle nových souřadnic.
POZOR : v uživatelem definovaných souřadnicových systémech se mohou objevit úhly zkreslené.

Ovládání animace

turtle.delay(delay=None)

zpoždění – kladné celé číslo
kreslení *Nastavte nebo vraťte zpoždění* v milisekundách. (Toto je přibližně časový interval mezi dvěma po sobě jdoucími aktualizacemi plátna.) Čím delší je zpoždění kreslení, tím pomalejší je animace.

turtle.tracer(n=None, delay=None)

n – nezáporné celé číslo
zpoždění – nezáporné celé číslo
Zapněte/vypněte animaci želvy a nastavte zpoždění pro aktualizaci výkresů. Li *Je dáno n* , ve skutečnosti je pouze každá n-tá pravidelná aktualizace obrazovky provedeno. (Lze použít k urychlení kreslení složitých grafika.) Při volání bez argumentů vrátí aktuální uložená hodnota n. Druhý argument nastavuje hodnotu zpoždění (viz delay()).

turtle.update()

Proveďte aktualizaci TurtleScreen. K použití, když je tracer vypnutý.
Viz také metoda RawTurtle/Turtle speed().

Použití událostí na obrazovce

turtle.listen(xdummy=None, ydummy=None)

Nastavte zaměření na TurtleScreen (za účelem shromažďování klíčových událostí). Hloupé argumenty jsou poskytovány, aby bylo možné projít listen()na metodu onclick.

turtle.onkey(fun, key)
turtle.onkeyrelease(fun, key)

fun – funkce bez argumentů resp None
klíč – řetězec: klíč (např. „a“) nebo klíčový symbol (např. „mezera“)
Spojte *zábavu* s událostí uvolnění klíče. Pokud *zábava* je None, vazby událostí jsou odstraněny. Poznámka: Aby bylo možné registrovat klíčové události, TurtleScreen musí mít zaměření. (Viz metoda listen().)

turtle.onkeypress(fun, key=None)

fun – funkce bez argumentů resp None
klíč – řetězec: klíč (např. „a“) nebo klíčový symbol (např. „mezera“)
Spojte *zábavu* se stisknutím klávesy, pokud je klávesa dána, nebo na jakoukoli událost stisknutí klávesy, pokud není zadána žádná klávesa. Poznámka: Aby bylo možné registrovat klíčové události, TurtleScreen musí mít zaměření. (Viz metoda listen().)

turtle.onclick(fun, btn=1, add=None)
turtle.onscreenclick(fun, btn=1, add=None)

fun – funkce se dvěma argumenty, která bude volána s souřadnice klepnutého bodu na plátně
btn – číslo tlačítka myši, výchozí 1 (levé tlačítko myši)
přidat – Truenebo False– pokud True, bude nová vazba přidáno, jinak nahradí dřívější vazbu
Spojte *zábavu* s událostmi kliknutí myší na této obrazovce. Pokud *zábava* je None, existující vazby jsou odstraněny.
Příklad pojmenované instance TurtleScreen screena instance Turtle jmenoval turtle:

turtle.ontimer(fun, t=0)

fun – funkce bez argumentů
t – číslo >= 0
Nainstalujte si časovač, který *milisekundách vyvolá zábavu* po *t* .

turtle.mainloop()
turtle.done()

Spustí smyčku událostí - volání funkce hlavní smyčky Tkinter. Musí to být poslední příkaz v grafickém programu želv. Nesmí *se* použít, pokud je skript spuštěn z IDLE v režimu -n (Žádný podproces) - pro interaktivní použití želví grafiky.

<u>Metody zadávání</u>

turtle.textinput(title, prompt)

název – řetězec
výzva – řetězec
Vyskočí dialogové okno pro zadání řetězce. Název parametru je nadpis dialogového okna, výzva je text převážně popisující jaké informace zadat. Vraťte vstup řetězce. Pokud je dialog zrušen, vraťte se None.

turtle.numinput(title, prompt, default=None, minval=None, maxval=None)

název – řetězec
výzva – řetězec
výchozí – číslo (volitelné)
minval – číslo (volitelné)
maxval – číslo (volitelné)
Vyskočí dialogové okno pro zadání čísla. titul je název dialogové okno, výzva je text většinou popisující jaké číselné informace zadat. default: výchozí hodnota, minval: minimální hodnota pro vstup, maxval: maximální hodnota pro vstup. Zadané číslo musí být v rozsahu minval .. maxval, pokud jsou daný. Pokud ne, vydá se nápověda a dialog zůstane otevřený oprava. Vraťte zadané číslo. Pokud je dialog zrušen, vraťte se None.
Nastavení a speciální metody

turtle.mode(mode=None)

režim – jeden z řetězců „standard“, „logo“ nebo „world“

Nastavte režim želvy („standardní“, „logo“ nebo „svět“) a proveďte reset. Pokud režim není zadán, je vrácen aktuální režim.

Režim „standardní“ je kompatibilní se starým turtle. Režim „logo“ je kompatibilní s většinou želvích grafik Logo. Režim „svět“ používá uživatelsky definovaný „světové souřadnice“. Pozor : v tomto režimu se úhly zdají zkreslené, pokud x/ypoměr jednotek se nerovná 1.

turtle.colormode(cmode=None)

cmode – jedna z hodnot 1,0 nebo 255

Vraťte barevný režim nebo jej nastavte na 1,0 nebo 255. Následně hodnoty r, g, b barevných trojic musí být v rozsahu 0..*cmode*.

turtle.getcanvas()

Vraťte plátno této TurtleScreen. Užitečné pro zasvěcené, kteří vědí, co dělat udělat s Tkinter Canvasem.

turtle.getshapes()

Vrátit seznam jmen všech aktuálně dostupných tvarů želv.

turtle.register_shape(name, shape=None)

turtle.addshape(name, shape=None)

Přidejte tvar želvy do seznamu tvarů TurtleScreen. Pouze takto registrované tvary lze použít zadáním příkazu shape(shapename).

turtle.turtles()

Vraťte seznam želv na obrazovku.

turtle.window_height()

Vraťte výšku okna želvy.

turtle.window_width()

Vraťte šířku okna želvy.

Metody specifické pro Screen, které nejsou zděděny z TurtleScreen

turtle.bye()

Zavřete želvové okno.

turtle.exitonclick()

Svázat bye()způsob kliknutí myší na obrazovku.

Pokud je hodnota „using_IDLE“ v konfiguračním slovníku False(výchozí hodnota), zadejte také hlavní smyčku. Poznámka: Pokud je NEČINNÁ s -npřepínač (žádný podproces), tato hodnota by měla být nastavena na Truev turtle.cfg. V tomto případě je vlastní hlavní smyčka IDLE aktivní také pro klientský skript.

turtle.setup(width=_CFG['width'], height=_CFG['height'], startx=_CFG['leftright'], starty=_CFG['topbottom'])

Nastavte velikost a polohu hlavního okna. Výchozí hodnoty argumentů jsou uloženy v konfiguračním slovníku a lze je změnit pomocí a turtle.cfgsoubor.

šířka – pokud je celé číslo, velikost v pixelech, pokud plovoucí, zlomek obrazovka; výchozí je 50 % obrazovky

výška – pokud je to celé číslo, výška v pixelech, pokud plovoucí, zlomek obrazovka; výchozí je 75 % obrazovky

startx – pokud je kladný, počáteční pozice v pixelech zleva okraj obrazovky, pokud je negativní od pravého okraje, pokud None, středové okno vodorovně

starty – pokud je kladná, počáteční pozice v pixelech shora okraje obrazovky, pokud je negativní od spodního okraje, pokud None, středové okno svisle

turtle.title(titlestring)

titlestring – řetězec, který je zobrazen v záhlaví želvy grafické okno

Nastavit titulek okna želvy na *titlestring* .

turtledemo— Demo skripty

The turtledemobalíček obsahuje sadu demo skriptů.
Demo skripty jsou:

název	**Popis**	**Funkce**
bytedesign	komplexní klasické grafický vzor želvy	tracer(), zpoždění, update()
chaos	grafy Verhulstovy dynamiky, ukazuje ten počítač výpočty mohou generovat výsledky někdy proti očekávání zdravého rozumu	světové souřadnice
clock	analogové hodiny ukazující čas vašeho počítače	želvy jako hodiny ruce, časovač
colormixer	experimentujte s r, g, b	ondrag()
forest	3 stromy na šířku	randomizace
fractalcurves	Hilbertovy & Kochovy křivky	rekurze
lindenmayer	etnomatematika (indické kolamy)	L-systém
minimal_hanoi	Hanojské věže	Obdélníkové želvy jako Hanojské disky (tvar, tvar)
nim	zahrajte si klasickou hru nim se třemi hromadami tyčinek proti počítači.	želvy jako nimsticky, řízená událostí (myš, klávesnice)
paint	super minimalistické kreslící program	onclick()
peace	základní	želva: vzhled a animace
penrose	aperiodické obklady s draci a šipky	stamp()
planet_and_moon	simulace gravitační systém	složené tvary, Vec2D
round_dance	tančící želvy rotující párově naproti směr	složené tvary, klon tvarovat, naklánět, get_shapepoly, aktualizace
sorting_animate	vizuální demonstrace různé způsoby třídění	jednoduché zarovnání, randomizace
tree	(grafická) šířka první strom (pomocí generátorů)	clone()
two_canvases	jednoduchý design	želvy na dvou plátna
wikipedia	vzor z wikipedie článek o želví grafice	clone(), undo()
yinyang	další elementární příklad	circle()