

# STL Ångloksscript

Detta dokument är till för ångloksbyggare som vill bygga ånglok med hytter som använder sig av STLs script.

Jag utgår från att läsaren har en grundläggande förståelse för hur hyttbygge går till i Trainz, samt en grundläggande koll på Trainz scriptspråk.

All logik i STLs ångloksscript är uppdelat i två script, ett för loket och ett för hytten. Dessa är skrivna från grunden och ärver inte Auran/N3v:s DefaultSteamCabin-script. Det finns dock vissa likheter. Scripten tillhandahåller standardimplementationer av de vanligaste reglagen i en svensk ånglokshytt, men är samtidigt tänkta att öppna för möjligheten att byggas vidare med individuella lösningar för varje lok. Till exempel finns bara en grundläggande implementation av belysningsfunktionaliteten för att varje lok ska kunna implementera sin egen belysning, oberoende av om det är gasbelysning, elektrisk belysning eller hur denna belysning faktiskt styrs från hytten. Hyttscriptet kräver att även loket det används till har STLs hyttscript för ånglok, annars kommer ett exception att kastas vid uppstart.

## Inbyggd funktionalitet

- Grundläggande reglage som regulator och omkastare, sotare.
- Bromventil av både system Knorr och New York.
- Lokbroms av både direktverkande trycklufts- och ångbromstyp.
- Hastighetsmätare system Deuta/Penta.
- Manometrar konverterade till kg/cm<sup>2</sup>.
- Hyfsat realistisk implementation av Greshams injektor.
- Realistiska vattenståndsglas med fysikaliska finesser som missvisning pga. banans lutning.
- Fungerande ventilställ som påverkar de olika ångbaserade komponenterna.
- Sandning med visuella effekter och möjlighet till sandning bakom hjulen på tanklok.
- Sotare och utblåsningsventiler med rök- och ljudeffekter.
- "Spegling" av hyttens dörrar och fönsters öppningsgrad på den exteriöra modellen.
- Stöd för rökeffekter påverkas av lokets pådrag, fyr etc.

## Att lägga till i configen

Först måste du lägga till STLs scriptbibliotek till script-include-table:

```
script-include-table
{
    mscommonsource                <kuid2:177292:209000:3>
}
```

Detta gäller för både hytter och ånglok som skall använda sig av STLs ångloksscript.

Dessutom skall dessa två rader in i kuid-tablen för loket, vilka redan bör vara där om loket redan har STL-koppel:

```
mstrainmonitor                <kuid:177292:204091>
mscommonsource                <kuid2:177292:209000:3>
```

I hyttens kuid-table skall dessa två rader in:

```
mscommonsource                <kuid2:177292:209000:3>
controlset                    <kuid2:328014:3200:1>
```

Sedan ska naturligtvis ånglokets/hyttens script ärva ifrån STLs script istället för Auran/N3Vs. De klasser som ska ärvas heter **STLSteamEngine** respektive **STLSteamCabin**. Ingen specifik kod behövs implementeras i klassen för att saker ska börja fungera, men se de bifogade templete-scriptet för detaljer.

## De inbyggda reglagenamnen i hytten

Här är en lista på namn de inbyggda reglagen, mätarna och annat i hytten som finns inbyggt i STLs ångloksscript. Genom att namnge en mesh i interiorns mesh table till dessa namn och låta interiorns script ärva från STLs ånglokshyttscript kommer dessa reglage att börja fungera automatiskt. I listan anges limits och ibland även notches som syftar på taggarna i interiorns mesh table vilka måste ha de värdena som anges för att fungera ordentligt med scriptet. Oftast är limits 0,1 och då är 0 stängt och 1 öppet läge om ej annat anges i förklaringen.

### Mätare

#### deuta\_speedo

Hastighetsmätare system Deuta/Penta, som visar hastigheten och skiljer på framåt/bakåt.

I configen ska det stå till exempel limits -20, 33.333 (i m/s) om hastighetsmätaren går mellan 120 km/h framåt till max 72 km/h bakåt.

Se till att visaren går åt höger (moturs) vid framåtgång och vänster (medurs) vid backgång. Om den går fel håll, byt från positivt värde till negativt värde eller tvärtom för de båda värdena i angles i configen.

#### main\_reservoir\_needle

Manometer för huvudbehållaren, i kg/cm<sup>2</sup> övertryck. I configen ska alltså mätaren ha limits mellan 0 till maxtrycket i kg/cm<sup>2</sup> som manometern är graderad till.

brake\_cylinder\_needle

Manometer för lokets bromscylinder, i  $\text{kg/cm}^2$  övertryck. Limits som ovan. Se även loco\_steam\_brake\_lever för specialfall när ångbroms används.

brake\_pipe\_needle

brake\_pipe\_needle1

Två manometrar för huvudledningen i  $\text{kg/cm}^2$  övertryck. Limits som ovan.

boiler\_needle

boiler\_needle1

Två manometrar för panntrycket i  $\text{kg/cm}^2$  övertryck. Limits som ovan.

steam\_chest\_needle

Manometer för trycket i ånglådan i  $\text{kg/cm}^2$  övertryck. Limits som ovan.

steam\_heat\_needle

Manometer för ångvärmeledningstrycket i  $\text{kg/cm}^2$  övertryck. Limits som ovan.

## Reglage

regulator

reverser

firebox

fire\_plates

Dessa reglage fungerar och namnges som de gjort tidigare enligt Auran/N3Vs standard. Motsvarar regulator, omkastare och fyrbox och fyrboxlucka.

whistle\_lever

Spak till ångvissla. Limits 0,1.

knorrbrake\_lever

Knorr förarventil (dock i Trainzutförande) Bromsen ansätts gradvis genom att byta till application och sedan lap. Tekniskt sett samma som den tidigare "trainbrakelap\_lever".

newyorkbrake\_lever

New York-broms förarventil med givna trycksänkningslägen. Bromsventil med flera förutbestämda bromslägen där trycket i huvudledningen automatiskt sänks till ett givet värde. Tekniskt sett samma som den tidigare "trainbrake\_lever". Se Aurans dokumentation för exempel på configvärden.

### loco\_steam\_brake\_lever

Ångbromsventil (broms verkande enbart på loket) Har tre lägen, 0=lossar bromsen, 1=mittläge, 2=ansätter bromsen. I configen ska reglaget alltså ha notches 0, 0.5, 1 och limits 0,2

Trainz kommer att hantera ångbroms på samma sätt som direktverkande tryckluftbroms. Enda skillnaderna i scriptet är att ångbromsen skickar en signal till loket att starta rökeffekten för att släppa ut ånga samt att ångbromsen ej kan användas då ventilstället är avstängt. Hyttskaparen får själv se till att ångbromsmanometern visar lämpliga värden vid maxtryck genom att modifiera limits på brake\_cylinder\_needle.

### loco\_air\_brake\_lever

Direktbromsventil system Knorr (broms verkande enbart på loket) för lok med tryckluftsbroms. Samma lägen som loco\_steam\_brake\_lever.

### handbrake\_lever

Handbromsskruv. Ska enbart användas som "parkeringsbroms". Fungerar för närvarande inte, men kan komma att fixas i framtiden. I config limits 0,1. (Där 0 ska föreställa lossad broms och 1 fullt tillsatt)

### blower

Ventil för hjälpblästern/sotaren. Limits 0,1.

### pysare\_lever

Spak för utblåsningsventilerna "pysarna". Limits 0,1 notches 0,1.

### sanding\_valve

Ventil för sandningen med två flödesnivåer och stöd för sandning framför eller bakom drivhjulen. För tenderlok limits 0, 2 och 3 notches. För tanklok limits -2, 2 och 5 notches. Här kommer 0 vara mittläget (ingen sandning) 1 vara lite sand och 2 mycket sand. Minus betyder sandning i backgång (sandning bakom drivhjulen).

Scriptet kommer att ta hand om att både dessa typer av sandningsventiler har samma namn, de skiljs med hjälp av limits och notches.

### reverser\_lock

Låsspak till omkastaren. Måste vara upplyft för att omkastaren ska kunna flyttas med musen. Påverkar inte tangentbordsstyning med F och R. I configen är 0 låst läge och 1 upplåst läge. Limits 0,1 notches 0,1.

### damper\_front

### damper\_back

Främre respektive bakre damp. Limits 0,1 där 0 är helt stängd och 1 är helt öppen i valfritt antal notches (eller utan notches om de kan ställas in steglöst). Påverkar mörkheten på röken från loket. Påverkar ej lokets förbränning.

## Vattenståndsglas

Vattenståndsglas av SJ standardmodell under 1900-talet med tre kranar för avstängning av övre respektive nedre förbindelsen med pannan, samt avloppsrör för genomblåsning av glaströret.

### waterglass\_left

Vänster vattenståndsglas vattennivå. Animerad vattenpelare i vattenståndsglasets. Ska se ut precis som enligt Aurans/N3Vs standard i configen. Limits 0, 100.

### waterglass\_left\_top\_tap

Övre avstängningskran för vattenståndsglasets. Limits 0,2, notches 0,0.5,1. Där 0 är arbetsläget (i regel 45 grader nedåt), 1 är avstängt läge (rakt ut) och 2 är genomblåsning (45 grader uppåt).

### waterglass\_left\_bottom\_tap

Nedre avstängningskran för vattenståndsglasets. Limits 0,1, notches 0,1 där 0 är arbetsläge (45 grader nedåt) och 1 är avstängt läge (45 grader uppåt).

### waterglass\_left\_drain\_tap

Den kran till avloppsröret som sitter längst ned på vattenglashuset vilken öppnas när glaset ska genomblåsas. Limits 0,1 där 0 är stängd och 1 är öppen.

Höger sida är samma grej, men med "right" istället för "left" i namnet.

Här kan man välja om man vill ha enkla eller avancerade vattenståndsglas. Om man bara vill visa vattenståndet använder man bara waterglass\_left/waterglass\_right kontrollerna och har inte med kranarna. Vill man ha avancerade vattenståndsglas tar man med kranarna. Scriptet kräver att man antingen har alla eller inga kranar med, annars slängs ett exception från hytten när locket initieras.

Vattenståndsglasen missvisar automatiskt vattennivån när locket körs i backar baserat på banans lustning. För att detta ska bli korrekt så måste variabeln *halfBoilerWaterMirrorLength* sättas till ett korrekt värde i hyttens script. Den har ett standardvärde som passar för B-lok, men kan behöva ändras för mindre lok. Se template-scriptet för hytten för detaljer.

## Injektorer

Injektorer av Greshams modell. Borde även gå att använda till Friedmanns och andra injektorer med samma uppbyggnad med vatten- och spillrör.

### injector0

Ångpådrag till eldarens injektor. Limits 0,1.

### injector0\_water\_tap

Avstängningskranen till eldarinjektorns rör till vattentanken. Limits 0,1 där 1 är öppen. Har locket denna så sätts den automatisk som öppen (1) vid start.

### injector0\_overflow\_tap

Avstängningskranen till eldarinjektorns spillrör. Limits 0,1, notches 0,1. Har locket denna så sätts den automatisk som öppen (1) vid start.

För förarinjektorn gäller samma namn, men med en 1:a istället för en 0:a i namnet.

Injektor 0 ska vara eldarens injektor och 1 förarens, då 0 är standardinjektorn i Trainz och i verkligheten är det eldarens injektor som används mest.

Injektorerna kan liksom vattenståndsglasen väljas som enkla eller avancerade. Om man vill ha enkla injektorer använder man bara injector0/injector1 och då kommer de fungera precis som på andra ånglok. Vill man ha avancerade måste alla fyra \*\_tap-controllers vara med. Om någon saknas slängs ett exception när hytten skapas.

## Ventiler

### crown\_main\_valve

Huvudventilen för ventilstället/kronan. Oftast samma som snabbavstängningen på 1900-talsånglok. Kommer att stänga av de saker som är kopplade till kronan. Limits 0,1 . 0 är normalläget (ventilen till ventilstället öppet) och 1 är utlöst läge (ventilstället avstängt och alla saker kopplade till ventilstället kommer att vara avstängda).

### air\_pump\_valve

Avstängningsventil för luftpumpen. Limits 0,1. (Ej kopplad till ventilstället som standard) Sätts automatiskt som öppen vid start.

### steam\_heat\_valve

Ventil för ångvärmen på ventilstället. Påverkar ångvärmemanometern. (Påverkar dock ej ångmängden i ångpannan) Limits 0,1.

### steam\_brake\_main\_valve

Ventil för avstängning av ångan till ångbromsen uppe på ventilstället. Limits 0,1, notches 0,1. Sätts automatiskt som öppen vid start.

### turbo\_generator\_valve

Avstängningskran på ventilstället för turbogenerator system Pyle/Sunbeam/ASEA-Stal, Limits 0,1. Är stängd vid start.

Alla dessa ventiler är valfria. Om locket inte har någon crown\_main\_valve, så kommer de andra ventilerna som är kopplade till kronan att fungera som att ventilen till kronan är öppen.

## Annat

Följande namn används på föremål i hytten som ska "spegla" sin animation på den exteriöra modellens motsvarande mesh, vilken ska ha exakt samma namn i lokets mesh table. Används till

exempel för att kunna öppna dörren i hytten och få dörren på den exteriöra meshen att öppnas lika mycket. Antalet frames i animationerna behöver inte överensstämja mellan hyttens och den exteriöra modellen, de mäts i "procent öppenhet". 0 är standardläget och 1 motsatt läge. Hur de definieras är upp till hyttbyggaren, men den exteriöra och interiöra animationen måste ha samma standard- och motsatta lägen, annars blir speglingen bakvänd. På detta sättet kan byggaren välja om till exempel skyddsglasen ska vara ut- eller infällda vid start.

door\_left

door\_right

Vänster/höger dörr. Limits 0,1.

door\_window\_left

door\_window\_right

Vänster/höger dörrfönster. Limits 0,1.

side\_window\_left

side\_window\_right

Vänster/höger skjutfönster. Limits 0,1.

window\_front\_left

window\_front\_right

Vänster/höger främre öppningsbara fönster. Limits 0,1.

window\_back\_left

window\_back\_right

Vänster/höger bakre öppningsbara fönster. De flesta tenderlok med öppen hytt saknar dessa (eller så är de en del av tendern). Limits 0,1.

sight\_glass\_front\_left

sight\_glass\_front\_right

sight\_glass\_back\_left

sight\_glass\_back\_right

Vänster/höger främre/bakre skyddsglas (De som skyddar lokpersonalen mot farvinden när de

hänger ut genom sidofönstret) På de flesta tenderlok används enbart de två front-varianterna. Limits 0,1.

roof\_vent

Taklucka. Limits 0,1.



# Rökeffekter

Scriptet har stöd för flera typer av rökeffekter på loket. Vissa av dessa är obligatoriska, men de flesta är valfria. En del typer som till exempel utblåsningsventilerna har stöd för att styra valfritt antal emitters (smoke-containers i lokets config), medan andra, som turbogeneratorns avlopp bara kan styra en emitter.

En del av rökeffekternas attribut, så som storlek, hastighet och ljushet ändras av scriptet. Gränserna för detta kan ställas in, se nedan.

När jag pratar om id:n syftar jag på det unika tal som rökeffektern har i configen. Till exempel smoke5 har id 5.

## Obligatoriska effekter/effekter med fasta id:n

### smoke0

Den generella ångplymen från skorstenen av avloppsångan från cylindrarna. Den är påslagen när det finns tryck i cylindrarna. Färgen på denna och storlek styrs från scriptet och kan ställas in, se nedan.

### smoke1

Tänkt att användas som en statisk väldigt subtil rök från skorstenen som alltid är påslagen. Detta id är dock inte kopplat på något sätt till scriptet.

### smoke2

Ånga från sotaren som kommer upp ur skorstenen. Storlek och färg styrs av scriptet och kan ställas in, se nedan.

### smoke3

Läckage från förarens injektor genom spillröret när injektorn suger. Bör vara en väldigt liten ström av ånga, knappt märkbar. Kan stängas av om den inte önskas, se nedan.

### smoke4

Samma som smoke 3, fast på eldarsidan.

### smoke5

Kraftig ångström från förarens injektor när injektorn spiller. Används dels när injektorns ångventil är öppen, men kranen till vattenröret är stängt och dels en kort stund när injektorn startas och stängs av och ångtrycket är för lågt för att öppna backventilen in till pannan. Av och påstängning styrs av scriptet så det ska bara vara en loopande emitter.

### smoke6

Samma som smoke 5, fast på eldarsidan.

## Frivilliga rökeffkter/effekter med valbara id:n

Många av rökeffekterna är frivilliga. Vissa kanske bara finns på en del lok och andra kan variera i

antal. Dessa måste pekats ut i lokets egna script genom att sätta ett par variabler i lokets init-metod. Det finns två varianter på variabler, antingen är de integers eller strängar. På integers kan man bara ha ett id, alltså bara en rökemitter. På de som är strängar kan man däremot valfritt antal rökeffekter användas. Integers har defaultvärde -1 vilket markerar att de inte används. Sätts ett positivt heltal så används motsvarande smoke-effect från lokets config.

Stängarna ska vara på samma form som den stäng som skickas som minor när röken ska slås till, dvs alla smoke-idn med + tecken framför varje. Till exempel "+10+11+12" om rökeffekterna 10, 11 och 12 ska användas. Lämnas strängen tom, "" så stängs den effekten av.

Alla smoke-effects ska vara av typ speed och alltså normalt loopas. Scriptet sköter själv om att stänga av och slå på dem. Eventuellt kan typ anim användas för att synka med drivhjulens animation.

Följande variabler finns:

#### SMOKE\_ID\_STEAM\_HEAT\_SAFETY

(integer) Den rökeffekt som slås på när säkerhetsventilen till ångvärmen blåser. Sitter normalt som ett litet rör som sticker upp genom taket på loket, vid sidan om visslan.

#### SMOKE\_ID\_TURBO\_GENERATOR

(integer) Avloppsröken från turbogeneratoren. Slås på när turbogeneratoren slås på.

#### SMOKE\_ID\_STEAM\_BRAKE

(integer) Avloppsånga från ångbromsen. Effekten körs under en viss tid och stängs sedan av. Längden styrs av scriptet och beror på hur högt tryck det var i bromscylindern. Normalt ett rör som sitter under hytten riktat nedåt.

#### SMOKE\_ID\_PYSARE\_ON

(string) Alla de smoke effects som ska slås på när utblåsningsventilerna blåser. Storlek och hastighet styrs av scriptet, se nedan.

#### SMOKE\_ID\_SAND\_FRONT\_ON

(string) De smoke effects som ska slås på när sandning sker framför drivhjulen, alltså då loket körs framåt.

#### SMOKE\_ID\_SAND\_BACK\_ON

(string) De smoke effects som ska slås på när sandning bakom drivhjulen sker, alltså vid backgång. Används normalt inte på tenderlok.

#### SMOKE\_ID\_CYLINDER\_LEAK\_ON

(string) Läckage från cylindrar. Slås på när det finns ånga i cylindrarna. Här kan mode anim med fördel användas på smoke-effecten i lokets config för att så ångan att öka eller minska i intensitet med drivhjulsväret.

#### SMOKE\_ID\_AIR\_PUMP\_ON

(string) Avloppsrök och läckage från luftpumpen. Slås på under en kort stund när pumpen gör ett slag. Normalt har man en emitter i skorstenen för avloppet och en eller flera på själva pumpen för läckage från otäta packningar.

Om man inte vill att det ska komma ånga från injektorerna när de körs så kan man sätta SMOKE\_ID\_INJECTOR\_LEFT respektive RIGHT till -1 i init()-metoden på lokets script. Smoke3 respektive smoke4 kan därefter användas till andra rökeffekter.

## Rökeffekternas attribut

På en del av lokets rökeffekter styrs vissa attribut från STLs script. Typiska sådana attribut är storlek och hastighet som partiklarna emitteras. Även färgen på skorstensröken kan justeras. För att detta ska bli så generellt som möjligt kan man enkelt i varje loks script bestämma mellan vilka värden dessa attribut får justeras av scriptet.

För varje inställbart attribut finns två variabler, en vars namn slutar med *Base* och en vars namn slutar med *Factor*. För varje attribut så är *Base* det minsta värdet och *Factor* anger det maximala värdet som får läggas till *Base*. Det vill säga så antar attributet ett värde mellan *Base* upp till *Base+Factor*. Ifall ett attribut ska varieras mellan 0 upp till något värde så sätts *Base* till 0 och *Factor* till det maximala värdet. För attribut som man vill ha konstanta så väljs *Base* till det önskade värdet och *Factor* till 0.

`stackMaxSize*` (`stackMaxSizeBase` & `stackMaxSizeFactor`)

Styr den slutliga storleken på rökpartiklarna från skorstenen (smoke0). Storleken vid emission (alltså vid skorstenen) styrs med min-size i configen som vanligt. Storleken är beroende av trycket i cylindrarna.

`stackVelocity*` (`stackVelocityBase` & `stackVelocityFactor`)

Styr hastigheten som rökpartiklarna från skorstenen har när de emitteras i m/s. Hastigheten är sedan beroende av trycket i cylindrarna.

`blowerMinSize*` (enligt ovan...)

Storleken på sotarens rökpartiklar när de emitteras från skorstenen. Är beroende av sotarens pådrag.

`blowerMaxSize*`

Storleken som sotarens rökpartiklar får innan de försvinner. Är beroende av sotarens pådrag.

`blowerVelocity*`

Hastigheten som sotarens rökpartiklar emitteras med i m/s. Är beroende av sotarens pådrag.

`blowerLifetime*`

Livstiden för sotarens rökpartiklar. Är beroende av sotarens pådrag.

`pysareMaxSize*`

Den storlek som pysarnas rökpartiklar får innan de försvinner. Är beroende av trycket i cylindrarna.

`pysareVelocity*`

Den hastighet som pysarnas rökpartiklar emitteras med i m/s. Är beroende av trycket i cylindrarna.

Inställningarna av attributen görs lämpligen i Init()-metoden efter att **inherited()** anropats.

Alla dessa variabler har standardvärden som passar bra för de flesta relativt stora lok. Man behöver därför inte sätta dem för varje lok om default-värdena fungerar bra.

För röken ur skorstenen, både avloppsröken och sotar-röken kan färgen på rökeffekten styras. För detta finns fyra objekt, *blowerDark*, *blowerLight*, *stackDark* och *stackLight*. Dessa är av typen RGBA och är således ett objekt med röd-, grön-, blå- och alphakanal. Dessa kan sättas Init-metoden med följande sätt:

```
stackDark.SetValues(R, G, B, A);
```

där R, G, B och A är flyttal som talar om kanalens intensitet med ett värde mellan 0 och 255.

Scriptet kommer att interpolera mellan dark och light-färgerna beroende på bland annat damparnas lägen och fyrens temperatur.

## Ljudeffekter

På loket använder scriptet flera sound-events för att kunna starta och stänga av ljud. Det är upp till lokbyggaren att förse loket med alla de sound-events som den ska ha och lägga till dessa i soundscript-containern i lokets config. På detta sätt går det att ha olika ljud på olika lok. Men man kan också använda de standardljud som används på STLs ånglok.

### cylinder\_drain

(loop) Den ljudloop som slås på när utblåsningsventilerna blåser ut ånga.

### blower\_low

### blower\_med

### blower\_high

(loop) Tre ljudloopar som körs när sotaren går. De är olika kraftighet på ljuden, så *low* används när sotaren kört lätt, *med* när den körs mellankratigt och *high* när den körs nära fullt. Med fördel kan samma ljudfil användas, men med olika volym på ljudet som ställs in i soundscript-containern i configen.

### injector\_left\_on

(loop) Den ljudloop som körs när injektorn suger.

### injector\_left\_spill

(loop) Den ljudloop som körs när injektorn spiller. (fräsande ljud)

### injector\_left\_hiss

(Ej loop) Det ljud som låter när injektorn slås på eller av och spiller på grund av att trycket är för lågt för att öppna backventilen in till pannan. Använd taggen *repeat-delay 0.01* i soundscript-containern för att ljudet inte ska loopas. Kort fräsande/pysande ljud på ca 1 sek.

### injector\_right\_on

### injector\_right\_spill

### injector\_right\_spill

Samma som ovan, men för höger sida. Samma ljudfiler kan med fördel användas på båda sidorna, men båda behöver finnas i soundscript-containern i configen.

### steam\_heat\_safety

(loop) Den ljudloop som körs när maximala trycket för ångvärmeledningen överstigs och säkerhetsventilen går. Pysande ljud, likt det från de vanliga säkerhetsventilerna, men lugnare då trycket bara uppgår till 4.5 kg/cm<sup>2</sup>. Behövs ej på lok som saknar ångvärme.

### turbo\_generator

(loop) Den ljudloop som körs när turbogeneratoren används. Normalt ett högfrekvent

vinande/visslande ljud. Behövs ej på lok som saknar elektrisk belysning.

#### `air_pump`

(loop) Den ljudloop som körs när luftpumpen går. Motsvarar en hel cykel, vilket för enkla eller tvåstegs Knorrpumpar motsvarar ett dubbelt slag (ca 1.5 sek)

#### `steam_brake_release`

(Ej loop) Den ljud som låter när ångbromsen släpps. Ett ganska kraftigt pysande ljud på ca 1-3 sekunder. Använd taggen *repeat-delay 0.01* i soundscript-containern för att ljudet inte ska loopas. Behövs ej på lok med direktverkande tryckluftsbroms istället för ångbroms.

## Extra scriptfunktioner

För att underlätta för ångloksbyggare har STLs ångloksscript en del praktiska metoder som man kan använda.

### SteamThreadUpdate()

Genom att överrida SteamThreadUpdate()-metoden i scriptet till sitt ånglok kan denna användas för att uppdatera loket på olika sätt med jämna intervall. Metoden anropas med ca 2 sekunders intervall av STLs ångloksscripts uppdateringsloop när den uppdaterar alla rökeffekter. I denna metod kan enklare uppdateringar av loket göras. Den kan till exempel användas för att slå på eller av egna rök- och ljudeffekter. Notera att inga tyngre beräkningar bör göras i denna metod då det kan påverka prestandan negativt. Undvik att ha grejer här som kan uppdateras med event istället. (Ha till exempel inte checkar för att slå på rökeffekter som styrs med en ventil. Slå istället på dem när ventilen ställs om) Anropa aldrig denna metod från lokets script.

### Belysning

STLs ångloksscript har också funktionalitet för att styra belysning. Det finns stöd för både elektrisk och gas(/fotogen-)belysning. Dessa två metoder kan överridas i ånglokhyttens script:

```
void OnLightMainSwitchChanged()  
void OnLightChangedFromUI()
```

Glöm inte att anropa **inherited()** först i båda dessa metoder!

Dessa metoder anropas av STLs interna script då belysningen ändras. Den första anropas när huvudbrytaren ändras. Med huvudbrytare menas på ett lok med elektrisk belysning ångpådraget till turbogeneratoren. Har inte loket någon sådan kommer denna metod aldrig att anropas.

Den andra metoden anropas av STLs script för alla typer av lok så fort strålkastarna slås på eller av från UI, eller om helljuset slås på eller av från UI:t. I denna metod bör ånglokshyttsscriptaren se till knappar i hytten uppdateras.

När belysningen ändras av egna script i ånglokhytten så måste två variabler i *STLSteamCabinData* hållas uppdaterade. Dessa är:

```
bool cabinData.highBeamsOn  
bool cabinData.headlightOn
```

Dessa skall uppdateras så fort belysningen ändras med knappar i hytten. Dessa sätts automatiskt av STLs script innan OnLightChangedFromUI() anropas när ljuset ändrats från UI:t och kan därför användas för att ta reda på vad ljusbilden ändrats till.

## ControlSet

STLs ångloksscript har ett eget control set som låter användaren definiera egna tangentbordsknappkombinationer för att styra grejer i ångloken. För att dessa ska aktiveras måste följande tagg läggas till i ånglokhyttens config:

```
controlset <kuid2:328014:3200:1>
```

Denna kuid skall givetvis läggas till i kuid-tablen också.