Preparação da fase 2 dos mods de ônibus no Proton Bus Simulator

Este documento é un rascunho. Vai demorar um pouco para ficar mais abrangente. Por favor, tenha paciência, lembre-se de que este é um projeto indie feito no tempo livre. A melhor forma de entender como os mods são feitos é fuçando nos mods de exemplo. Boa parte dos nossos mods têm o 3D livre para edição e análise. Eles serão postados no site:
www.busmods.com

A fase 2 do sistema de mods de ônibus traz várias animações, que anteriormente estavam disponíveis apenas nos ônibus nativos. A maioria das coisas exige apenas colocar novas peças com nomes específicos no 3D. Algumas outras propriedades e comandos foram adicionados. Os mais significantes são explicados neste documento, enquanto os mais simples podem ser explorados diretamente nos mods de exemplo.

Para prosseguir com este guia você já precisa saber converter os mods na fase 1, senão poderá ficar sem entender as coisas. Se estiver iniciando do zero, por favor, veja os tutoriais da fase:
http://blog.protonbus.com.br/2018/09/primeira-fase-do-sistema-de-mods-de.html

Salvo onde indicado o contrário (portas, sons, luzes etc), a maioria das configurações são feitas no arquivo principal do ônibus.

Versão e tela de seleção do mod

```
[mod]
name=MiBRTS
busModVersion=2
preview=preview.png
```

Altere o item **busModVersion** para 2. Isto será útil no futuro, quando a fase 3 suportar novas coisas… O jogo atual lerá isso para emitir um aviso para os

jogadores de que o mod não é suportado (isso já ocorrerá se você testar com um 3 ou 5 ou qualquer número maior ali).

O item **preview** indica uma imagem para aparecer na tela de seleção. A imagem fica na pasta base do ônibus (a *baseDir*), não na *textures*. Se esse item for omitido, o jogo procurará automaticamente uma imagem chamada **preview.png**. Ou seja, se colocar a imagem com este nome, nem precisa colocar este item. Caso você faça vários ônibus usando a mesma pasta base para economizar espaço de texturas e sons, indique aqui o arquivo correto do veículo em questão.

Prefira usar a proporção 16:9 na imagem, como por exemplo, 1280x720 ou metade, para não ficar um arquivo grande: **640x360**.

Standard Shader

[config]
useStandardShader=1

0 = desativado, 1 = ativado

Esta opção força o uso do shader atual da Unity, o Standard Shader. O interior do ônibus pode ficar um pouco mais bonito, porém talvez fique mais pesado nos celulares. É uma opção que poderá ser removida se ficar pesada. Deixando em 0, ou omitindo, fica o padrão anterior (o shader Legacy/Diffuse) com cores mais lavadas e nenhum brilho/specular.

Como é uma opção experimental, ainda não é possível alterar os parâmetros do metallic, specular etc, eles ficarão no padrão 0.5.

Câmera de ré

[reverseCamera]
posX=0
posY=3.22161
posZ=-6.11213
rotX=30

rotY=180
rotZ=0

Comando similar ao espelho, só que no caso é uma câmera virada para trás, posicionada em algum lugar na traseira do ônibus. Pode ser levemente rotacionada para baixo no x. Note a rotação em 180 graus no eixo vertical (Y na Unity, Z no Blender), para ficar virada para trás. Nos espelhos retrovisores a base já fica virada no jogo, por isso não precisa rotacionar ali. Na câmera você deve especificar a rotação.

O monitor da câmera de ré é basicamente um plano, posicionado onde fica a tela. Ele ficará desativado no jogo, só aparecendo quando estiver na ré. Tem que ter um fundo embaixo dele, para não ficar um buraco enquanto não estiver na ré. O plano precisa ficar um pouco acima do fundo, para não ficar piscando (o tradicional Z-fighting nos games).

A peça da tela deve se chamar:

_reverse_camera_screen_

Ela tem que estar mapeada na textura inteira no UV, mas não precisa ter material nem textura associada. Diferente dos espelhos, não use o mirror no x no mapeamento, visto que a câmera não gera imagem invertida.

Fumaça do escapamento

[posSmoke1]
enabled=1
posX=-1.0966
posY=3.08716
posZ=-6.1505
rotX=0
rotY=0
rotZ=0
multiplier=0.5
colorR=0.2
colorG=0.2
colorB=0.2
colorA=0.1

O controle do escape ganhou novos parâmetros: o **multiplier** é um multiplicador de intensidade, para emitir mais ou menos partículas. Não exagere no valor para mais, pois pode pesar para os jogadores. Colocando 0.5 emite metade do que emitiria, colocando 2 emite o dobro, 1.2 um pouco mais, etc. Vá testando, mas não use números muito grandes como 5, 10 etc.

Os parâmetros color… definem a cor. Veja o próximo tópico sobre as cores para entender como elas são definidas.

Lá na seção **specialTextures** você pode definir a textura da fumaça que será usada no parâmetro **particleSystemSmoke1**. A imagem indicada aqui deve estar na pasta de texturas do ônibus. Em ônibus novos pode ser bom trocar para uma mais clara, além de ter alterado a cor. Se não definir uma textura, será usada a padrão do jogo. Veja os mods de exemplo, alguns podem vir com outras textura dessas. Experimente trocar para uma textura aleatória para ver o comportamento, caso você não sinta diferença.

[specialTextures]
particleSystemSmoke1=ParticleCloudWhite.png

Cores em várias configurações

Alguns comandos novos aceitam a personalização da cor. A engine usa cores no formato RGBA com valores de 0 (nulo) a 1 (cor total). O último elemento é o canal alpha, sendo 0 geralmente todo invisível e 1 opaco, onde um número intermediário define uma transparência. Geralmente o A vai ser 1 na maioria das cores, exceto em algumas para ficar mais apagadas (como o fundo da tela do marcador de ar condicionado, citado mais à frente).

Exemplos:

colorR=1
colorG=0
colorB=0
colorA=1

A cor acima fica vermelha, pois usou o valor máximo do R (red, vermelho), deixando o G (green, verde) e B (blue, azul) em 0. Se deixar o G em 1 e o R e B

em 0, fica verde. Se deixar o B em 1 e o R e G em 0, fica azul. A partir daí você pode ir misturando valores para chegar na cor desejada.

Há várias tabelas de cores na internet. A maioria cita os códigos hexadecimais, úteis em HTML, porém a engine do jogo precisa dos valores normalizados entre 0 e 1.

Você pode experimentar alguns valores desta tabela do site abaixo, ou procurar outras:

https://tug.org/pracjourn/2007-4/walden/color.pdf

Câmeras de comandos

Ao apertar C ou clicar no ícone das câmeras de comando no Android, o jogo altera a visão focada em áreas do painel onde o jogador pode clicar nos botões, ou apreciar algum instrumento informativo.

Essas posições de câmeras são identificadas pelo comando **command_camera_** seguido de um número incremental para cada posição. O jogador alterna entre eles apertando C e volta para a câmera padrão no F1 ou espaço.

[command_camera_1]
posX=-0.174367
posY=1.76852
posZ=5.27041
rotX=-62.1665d
rotY=-13.3138
rotZ=-0.000039

[command_camera_2]
posX=-0.344243
posY=1.85677
posZ=5.22591
rotX=-66.1773
rotY=-15.7203
rotZ=-0.00003

etc… Podem ter ilimitadas dessas câmeras. Alguns ônibus usam duas ou três, é só ir adicionando mais conforme necessário.

É bem demorado configurar elas, você precisará de bastante paciência. É uma coisa opcional porém desejável, especialmente para o controle de ar condicionado. Pode ser bom ver os vídeos, quando estiverem disponíveis.

Você precisa colocar um ponto onde a câmera será posicionada, como se estivesse olhando para o objeto. Depois imaginar a rotação que teria que fazer para focar no objeto (visto que a câmera padrão foca para a frente, como a cabeça do motorista). Rotacione negativamente em Y para virar para a esquerda, e positivamente para a direita. A rotação em X vai para cima e para baixo, e em Z vai para os lados.

A rotação no Blender pode não corresponder à mesma na engine, aí teria que ir aumentando ou diminuindo o valor até ficar bom, ou tentar inverter o sinal. Infelizmente cada programa usa um sistema de coordenadas incompatível entre si, sofremos muito com isso também.

Ar condicionado

O sistema de ar condicionado tem configurações espalhadas em vários arquivos, pode parecer confuso mas é bem simples, só precisa estar atento a todos os pontos. As definições e configurações da tela de comando ficam no arquivo principal. Os botões no modelo 3D, o som dele ligado no arquivo dos sons, com exceção do som da cortina de ar em cima das portas, que fica diretamente no arquivo das portas para facilitar o uso das coordenadas (geralmente vão ser as mesmas das luzes das portas, ou bem próximas).

O controle do ar condicionado é posicionado no arquivo principal do ônibus. Os comandos são estes:

```
[air_conditioner]
enabled=1
posX=-0.761969
posY=2.74968
posZ=5.72929
rotX=-18
rotY=0
```

rotZ=0
scaleX=0.0025
scaleY=0.0025
scaleZ=1

[air_conditioner_screen_on]
colorR=0
colorG=1
colorB=0
colorA=1

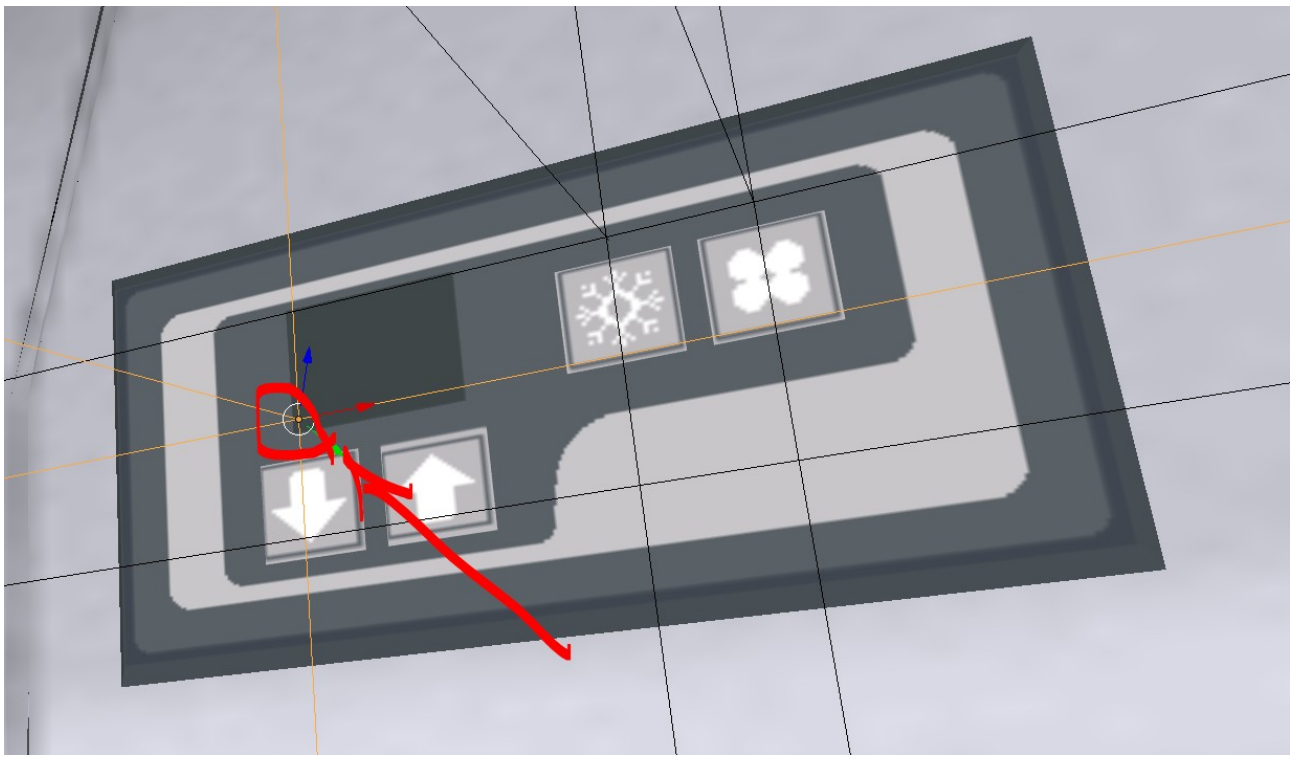
[air_conditioner_screen_off]
colorR=0
colorG=0.25
colorB=0
colorA=0.75

Destaque para o **enabled=1** no primeiro comando. Se deixar em 0 ele ficará desativado, não carregando os marcadores nem os sons, e ignorando os botões. Pode ser útil para desativar rapidamente, ou pelos próprios jogadores caso não gostem.

Os dois últimos definem a cor dos pontos da tela. Note o alpha na cor da tela para os pontos em off, para aparecerem com a cor porém um pouco mais transparentes.

Diferente da tela da câmera de ré, esta não precisa de um plano, o sistema gera ela dinamicamente na hora. Você indicará a coordenada do canto inferior esquerdo de onde ficaria a tela, e ajustará a escala em X e Y (a Z pode ficar em 1 aqui, pois será um plano 2D, sem profundidade).

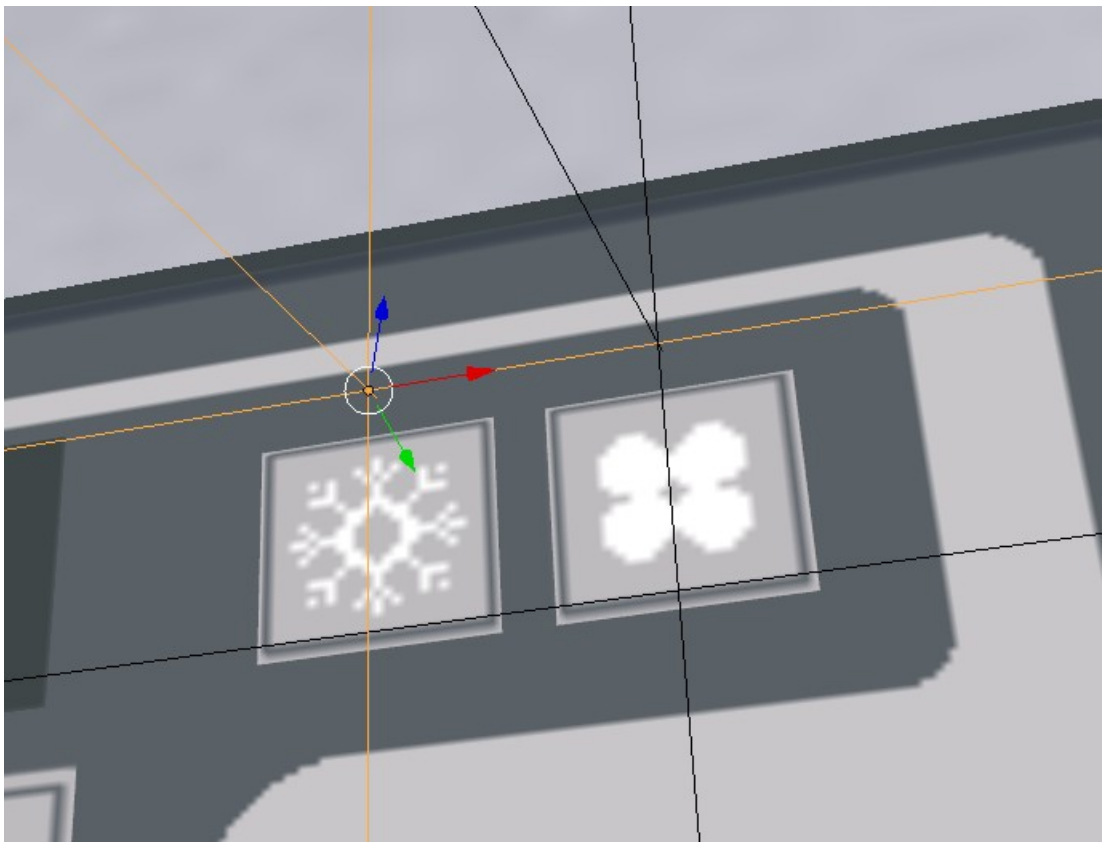
Coordenada para posição e rotação:



Observe o ponto destacado (o objeto empty para pegar a coordenada no Blender). Ele fica no canto inferior esquerdo da tela, um pouco para dentro da borda, não no meio.

Deixe ela um pouco para fora ou para cima do objeto, para que a tela não fique oculta nem piscando. A escala em 1 deixará um marcador gigante. Vá diminuindo o valor até ficar bom. Num dos mods de exemplo um valor legal foi 0.0025.

Acima dos botões de ar ligado e ventilador forçado, pegue mais duas coordenadas para pequenas luzinhas que ficarão acesas quando estes botões forem ativados.



Estas luzes serão definidas no arquivo das luzes (lights), assim:

[air_conditioner_on_1]
posX=-0.683482
posY=2.77392
posZ=5.72009
rotX=-18.6
rotY=
rotZ=
scaleX=0.01
scaleY=0.01
scaleZ=0.01
picture=Z_LuzAr.png

[air_conditioner_fan_1]
posX=-0.651946
posY=2.77392
posZ=5.72009
rotX=-18.6
rotY=
rotZ=
scaleX=0.01
scaleY=0.01

scaleZ=0.01
picture=Z_LuzAr.png

air_conditioner_on_1 é a do botão do ar ligado, e **air_conditioner_fan_1** do ventilador forçado, útil quando está muito calor.

Veja a seção de luzes com texturas extras neste documento para entender melhor sobre estes parâmetros. Vendo o mod de exemplo dá para entender bem também. A escala aumenta ou reduz o tamanho da luz. A rotação pode ser importante, especialmente no x, senão a luz ficará na vertical como ficam as lanternas traseiras.

Cada porta terá o som da cortina de ar diretamente no arquivo da porta (right ou leftdoor…). A estrutura é parecida com as dos sons tradicionais, porém ficando no arquivo da porta, para facilitar a configuração com as posições de uma por uma:

[doorsound_air]
enabled=1
file=sounds/door_air_cond.wav
posX=1.12496
posY=2.95696
posZ=-4.08266
minDist=1.5
maxDist=3
spatialBlend=1
volume=1
pitch=1

Para facilitar, geralmente dá para usar a mesma coordenada da luz da porta.

A configuração do minDist, maxDist e spatialBlend deve ser adequada para só tocar esse som perto da fonte (origem do som), evitando que o da porta do fundo seja ouvido muito alto lá na frente, e vice-versa. Geralmente isso é obtido deixando valores baixos em minDist e maxDist, e o spatialBlend em 1, para tornar o som 100% 3D (audível com relação à posição dele).

Os sons de ar das portas traseiras quase não dá para ouvir na posição do motorista, você pode deixar sem eles caso queira deixar o mod ainda mais otimizado. Muitos sons tocando junto podem ficar pesados, mas é legal ter para uma experiência mais completa (podem ser omitidos em mods “lite”).

No arquivo de som (sounds) devem ter duas referências para os arquivos de som do ar condicionado ligado. Estes são os sons mais importantes que geralmente abrangem o ônibus quase todo, ficando perto do equipamento do ar.

```
############################################################
################ AR CONDICIONADO FRACO
############################################################

[airConditioningSlow]
file=sounds/airConditioningSlow.wav
posX=0
posY=3
posZ=0
minDist=1
maxDist=5
spatialBlend=0.8
volume=0.5
pitch=1


############################################################
################ AR CONDICIONADO FORTE
############################################################

[airConditioningFast]
file=sounds/airConditioningFast.wav
posX=0
posY=3
posZ=0
minDist=1
maxDist=5
spatialBlend=0.8
volume=0.5
pitch=1
```

airConditioningSlow é o som dele ligado no estágio 1, e **airConditioningFast** no estágio 2. Veja a documentação ou os tutoriais de som, se disponíveis, para entender os demais parâmetros.

Para deixar o som mais alto, os parâmetros mais importantes são o minDist e maxDist, além do volume. O minDist é a distância mínima do som onde ele

ainda será ouvido no volume máximo, e conforme se distancia, entra o maxDist, onde ele ficará bem mais fraco ou inaudível.

A configuração spatialBlend deve receber um valor de 0 a 1, se deixar em 0 o som fica plano, sem simular posição no mundo 3D. Deixando em 1 ele fica no mundo tridimensional, ouvido apenas perto da origem (reduzindo o volume entre o minDist e o maxDist). Um valor intermediário faz com que ele seja percebido mais alto, porém se reduzir demais pode perder a noção 3D (ouvindo ele mesmo na câmera livre longe do veículo).

Por fim, as peças no 3D:

_airconditioner_button1_	botão de ligar/desligar o ar condicionado
_airconditioner_button2_	botão do ventilador forçado
_airconditioner_down_	botão para diminuir a temperatura
_airconditioner_up_	botão para aumentar a temperatura

Não é possível fazer todos os tipos de ar condicionados existentes no mundo, optamos por este que é bem genérico e atende a maioria dos casos. Se o ônibus desejado usar um sistema diferente, tente adaptá-lo ao padrão do jogo.

Reflexos

Os reflexos no Proton são definidos no arquivo de configuração, onde você altera os parâmetros metallic e glossiness do material. É um tema complexo, você pode ler mais sobre isso em manuais de criação de jogos digitais ou na documentação das próprias engines, como Unity ou Unreal.

Estes valores ficam entre 0 e 1. O parâmetro **metallic** define quão metálico é o material: 0 ele fica bem fosco, opaco, e 1 fica bem brilhante, reflexivo.

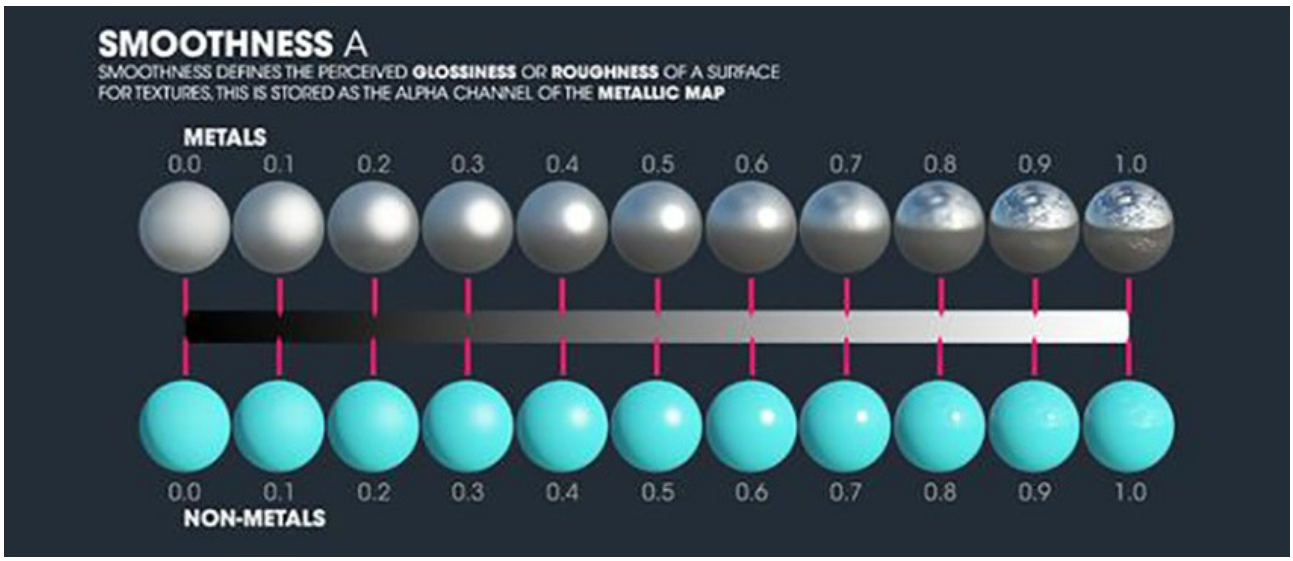


Imagem extraída da documentação da Unity, aqui:
https://docs.unity3d.com/Manual/BestPracticeMakingBelievableVisuals5.html

O parâmetro **glossiness** define a nitidez da imagem refletida, a grosso modo (não é bem isso, mas é quase). Deixando mais perto do 1, você verá melhor os detalhes do reflexo.

Um valor bom para alguns brilhos é deixar o metallic em torno de 0.3 a 0.6, e o glossiness mais forte, perto do 1. Se deixar o metallic perto de 1 ele pode ficar muito escuro, fugindo um pouco do objetivo principal.

Não é possível escolher a imagem a ser refletida, ela é padrão no Proton para todos os modelos e poderá ser substituída no futuro. Usamos uma imagem genérica para dar a sensação de reflexo, visto que um reflexo em tempo real fica bastante pesado nessa engine. Um sonho seria um reflexo real mostrando o ambiente ao redor, mas está fora dos planos por um bom tempo isso.

Você pode definir vários reflexos (por enquanto tem um limite de dez), e reutilizá-los em diferentes peças do seu ônibus. Por exemplo, pode criar o reflexo 1 para a carroceria deixando mais fosco, e um reflexo 2 para a máscara, deixando ela mais brilhante (útil em alguns modelos com a máscara grande na lateral). Isso não é só para o exterior, pode ser usado em peças do interior também, ou rodas. Ou até mesmo na carroceria, para diferenciar tampas e detalhes que precisam ser realçados.

//reflexos da lateral
[ref001]
metallic=0.4
glossiness=0.8

//reflexos da máscara
[ref002]
metallic=1
glossiness=1

etc…

Os nomes das peças que usarão os referidos reflexos devem ter o identificador deles usando o _, por exemplo, **_ref001_**. Lembre-se que os parâmetros podem ser combinados em alguns casos, por exemplo, *_ref001_skin001_carroceria*.

Vidros com reflexos

Os vidros com reflexos são criados de forma um pouco diferente. Este recurso é experimental. Tivemos muitos problemas com o Standard Shader com materiais transparentes nos mods, por isso utilizamos um shader alternativo. Se quiser deixar os vidros como na fase 1, sem reflexos, continue utilizando o comando _transparent_ e a vida segue normalmente. Futuramente isto poderá mudar.

Se quiser usar os reflexos em vidros, defina os vidros desta forma (há um limite de dez por enquanto):

```
//vidro externo
[glass001]
enabled=1
shininess=0.9
specColorR=0.3
specColorG=0.3
specColorB=0.3
specColorA=0
refColorR=0.2
refColorG=0.2
refColorB=0.2
refColorA=0
```

Depois nomeie a peça no 3D com o nome **_glass001_** etc.

Se deixar enabled=0 o vidro perde o reflexo, voltando a usar o modo antigo. Isso facilita os testes e permite que os usuários o desativem sem precisar mexer no 3D, já que é algo experimental e pode não agradar a todos.

O parâmetro **shininess** define o brilho, similar ao *metallic* do outro. Quanto mais próximo de 1, mais brilho.

Ele tem duas cores no vidro: specColor e refColor. A melhor forma de testar é colocando cores fortes diferentes e vendo no jogo. Aí você acha o tom adequado ao tipo de vidro que quer, se mais azulado, esverdeado, etc.

Deixando assim, por exemplo, uma ficará vermelha e outra azul, facilitando bastante ver no jogo o efeito:

```
specColorR=1
specColorG=0
specColorB=0
specColorA=0
refColorR=0
refColorG=0
refColorB=1
refColorA=0
```

Este tipo de reflexo não permite o uso de texturas muito escuras, elas ficarão claras. Um novo tipo de reflexo em vidro será estudado para versões futuras do jogo.

É importante testar tanto no modo de gráficos simples (usando o botão “baixo” na tela de opções) e depois no ultra, para ver como se comportaria em diferentes aparelhos. Às vezes um reflexo que ficou bonito no modo ultra fica bem feio no simples. É algo bem complexo mesmo.

Outra recomendação é separar o vidro interno, mantendo o interno sem reflexos, ou com bem poucos. Você pode manter o interno sem reflexos usando _transparent_ no nome dele, como antigamente, enquanto usa _glass001_ no novo. Ou então utilize outro glass… com um reflexo mais suave para a parte interna.

IMPORTANTE: nas peças de vidro com chuva, não utilize o reflexo! Ele deve ser usado somente nos vidros normais. Os vidros com chuva devem usar o *_transparent_*.

Falando em vidros, prefira usar peças diferentes para os vidros internos e externos, deixando uma textura mais clara no material do interior. Vemos muitos ônibus com vidros escuros por fora, mas por dentro, na vida real não é tão escuro. Se deixar a textura única para ambos, pode ficar muito feio por dentro, dificultando a visão do cenário e dos retrovisores. Note que o vidro do itinerário também é um pouco mais claro que os vidros externos da carroceria, nos modelos que possuem o visor do itinerário separado do vidro frontal.

Dica: como este comando glass… deixa o vidro muito claro, você pode duplicar a peça do vidro e utilizá-la com o comando **_transparent_** na cópia, ficando um com o glass e outro com o transparent. Desta forma teremos tanto um vidro escurecido, útil para a parte externa, como os reflexos.

Multiplex

Alguns ônibus modernos têm um painel eletrônico com botões suaves, que acendem e podem ter estágios diferentes de coloração: desligados, ligados, e com o comando ativo.

As texturas ficam definidas lá na seção **specialTextures**:

[specialTextures]
multiplexOff=MultiTextura.png
multiplexOn=MultiTextura.png
multiplexActive=MultiTextura_ON.png

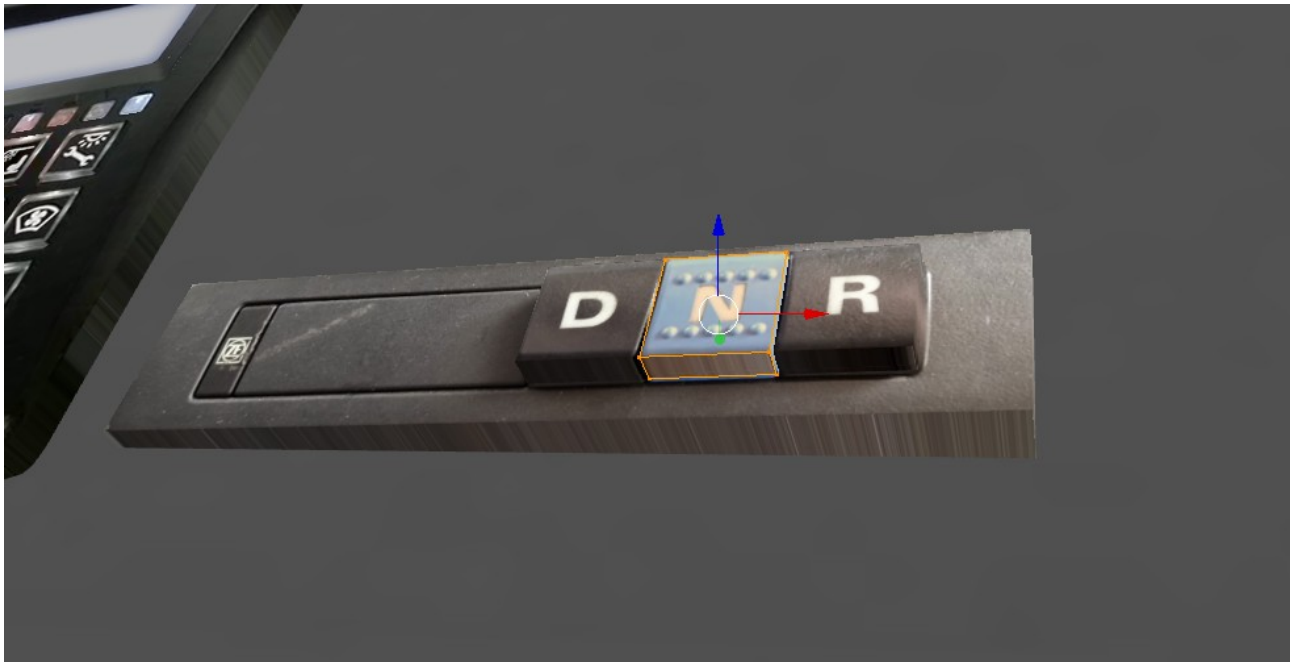
Você pode usar uma textura diferente em cada um dos estágios, ou repetir alguma delas, caso não tenha diferença na coloração.

Nem todos os botões do multiplex podem ser animados: cada ônibus tem uma configuração muito diferente do outro, não é possível suportar todos. Somente os botões genéricos mais comuns serão animados. Futuramente talvez mais botões serão suportados, mas não é algo prioritário, é um mero detalhe.

Câmbio

O câmbio no jogo pode ser escolhido pelo usuário, se deseja automático ou manual. A maioria dos ônibus funcionam com os dois no simulador, muito embora na vida real alguns são de um tipo só.

Para facilitar, é possível colocar ambas as peças dos manuais e automáticos, e o jogo alternará a exibição conforme a seleção do jogador. Basta nomear as peças com os nomes adequados.



No caso do câmbio automático, não são suportados aqueles Scania que têm 123DNR, o ativo será o 3 e não há previsão de mudar isso tão cedo devido a complexidade. A peça de plástico que é a base dos botões do câmbio automático mas não é botão também deve ficar separada com seu nome próprio, assim ela fica escondida quando a pessoa for jogar com o manual.

Os botões automáticos são clicáveis pelo jogador, já o manual não, ele apenas irá alternar de posição para dar um feedback visual. No momento são suportadas apenas 6 marchas + neutro + ré.

Luzes das portas

Estas seções serão atualizadas conforme der tempo… Fuce e explore os mods de exemplo para ir entendendo, por favor.

Luzes com texturas
Parâmetros de alguns materiais, cor pra destacar a carroceria ou linhas, parachoque etc, escurecer texturas etc.
Limpadores

Otimizadores de peças escondidas

Letreiros deslizantes

Animações personalizadas com estado on/off

_anim001_on off
_clickoo1_on off

EVITAR AUDIO EM MP3!

aviso das skins 2048

todos os paths de passageiros ou posições, no msm 3ds! Se dividir eles em 2 ou mais pode ter bugs inesperados