

Exports et événements

MasterLadder expose des **exports** et des **événements** pour que vos autres scripts (dispatch, HUD, scripts de feu, métiers...) sachent ce qui se passe sur les échelles. Le nom de la ressource doit rester `MasterLadder`.

Unités : angles en **degrés**, distances en **mètres**. Une échelle est identifiée par le **netId** de son véhicule.

Places

Code	Place
<code>seat</code>	Poste de commande (tourelle)
<code>b1</code> , <code>b2</code> , <code>b3</code>	Places de la nacelle
<code>c1</code> , <code>c2</code>	Sur l'échelle (montée à pied)

État d'une échelle (`GetLadderState`)

Les deux côtés renvoient la même table (ou `nil` si l'échelle n'est pas déployée) :

Champ	Contenu
<code>netId</code> , <code>model</code> , <code>label</code>	Véhicule, clé dans <code>Config.Ladders</code> , nom affiché
<code>phase</code>	calage automatique : <code>"deploy"</code> (calage), <code>"ready"</code> (manœuvrable), <code>"pack"</code> (repli) · calage manuel : <code>"setup"</code> (stabilisateurs au pupitre), <code>"unfold"</code> (dépliage), <code>"ready"</code> , <code>"fold"</code> (repli de la nacelle), <code>"abort"</code> (abandon, rangement automatique)
<code>calage</code>	<code>"full"</code> , <code>"reduced"</code> , <code>"left"</code> (gauche réduit) ou <code>"right"</code> (droit réduit)
<code>rotation</code> , <code>elevation</code> , <code>length</code> , <code>arm</code>	Position de l'échelle (°, °, m de longueur totale, °)
<code>basket</code>	Position de la nacelle (<code>vector3</code>)
<code>reach</code> , <code>maxReach</code>	Déport actuel et déport maximal de l'abaque (m)
<code>load</code>	Personnes comptées dans la charge
<code>occupants</code>	<code>{ seat = id, b1 = id, ... }</code> (0 = libre)

Champ	Contenu
<code>operator</code>	ID serveur de l'opérateur aux commandes (0 = personne)
<code>basketLocked</code> , <code>spotlight</code>	Verrou de nacelle, projecteur
<code>monitor</code>	<code>{ holder, flowing, yaw, pitch }</code> (+ <code>impact</code> côté client)

Côté client s'ajoutent `vehicle` (entité) et `height` (hauteur de la nacelle au-dessus du sol).

Exports client

```
local ml = exports['MasterLadder']
```

Export	Retour
<code>ml:GetPlace()</code>	<code>netId</code> , <code>place</code> du joueur, ou <code>nil</code>
<code>ml:IsInLadder()</code>	Le joueur occupe une place
<code>ml:IsInBasket()</code> / <code>ml:IsClimbing()</code> / <code>ml:IsAtSeat()</code>	Nacelle / sur l'échelle / au poste
<code>ml:IsOperator()</code>	Le joueur a les commandes
<code>ml:IsHoldingMonitor()</code>	Le joueur tient la lance
<code>ml:IsLadderVehicle(veh)</code>	Ce véhicule est une échelle configurée
<code>ml:IsDeployed(veh)</code>	Cette échelle est déployée
<code>ml:GetLadderState(netId)</code>	Table d'état (échelles affichées chez ce joueur)
<code>ml:GetBasketCoords(netId)</code>	Position de la nacelle
<code>ml:IsWaterFlowing(netId)</code>	La lance débite
<code>ml:GetWaterImpact(netId)</code>	Point d'impact du jet (<code>vector3</code>)
<code>ml:GetNearestLadder(range)</code>	<code>netId</code> , <code>distance</code> de l'échelle déployée la plus proche (défaut 50 m)
<code>ml:GetVisibleLadders()</code>	Liste des <code>netId</code> des échelles affichées

Exemple : couper un HUD de vitesse dans la nacelle.

```
CreateThread(function()
    while true do
        Wait(500)
        local inBasket = exports['MasterLadder']:IsInBasket()
        -- masquer / afficher votre HUD ici
    end
end
```

```
end)
```

Exports serveur

Export	Retour
<code>IsDeployed(netId)</code>	L'échelle est déployée
<code>GetLadderState(netId)</code>	Table d'état
<code>GetDeployedLadders()</code>	Liste des netId déployés
<code>GetBasketCoords(netId)</code>	Position de la nacelle
<code>GetPlayerPlace(src)</code>	<code>netId</code> , <code>place</code> du joueur, ou <code>nil</code>
<code>GetOccupants(netId)</code>	<code>{ place = src }</code> des places occupées
<code>IsWaterFlowing(netId)</code>	La lance débite
<code>EjectPlayer(src)</code>	Fait descendre le joueur de sa place (<code>true</code> si fait)
<code>EvictLadder(netId)</code>	Vide toutes les places et rend les commandes
<code>StopWater(netId)</code>	Ferme le jet (le porteur peut le rouvrir)

Événements serveur

À écouter avec `AddEventHandler` dans une ressource serveur :

Événement	Paramètres	Quand
<code>masterladder:deployed</code>	<code>netId</code> , <code>src</code> , <code>calage</code>	Déploiement lancé (<code>"full"</code> / <code>"reduced"</code> ; <code>"manual"</code> au début d'un calage manuel)
<code>masterladder:ready</code>	<code>netId</code>	Calage terminé
<code>masterladder:packing</code>	<code>netId</code> , <code>src</code>	Repli lancé
<code>masterladder:packed</code>	<code>netId</code>	Échelle rangée
<code>masterladder:ptoChanged</code>	<code>netId</code> , <code>src</code> , <code>engagée</code>	Prise de force engagée / désengagée
<code>masterladder:placeChanged</code>	<code>netId</code> , <code>src</code> , <code>place</code>	Place prise, ou quittée (<code>place = nil</code>)
<code>masterladder:controlChanged</code>	<code>netId</code> , <code>src</code>	Opérateur aux commandes (0 = personne)
<code>masterladder:monitorChanged</code>	<code>netId</code> , <code>src</code>	Porteur de la lance (0 = personne)
<code>masterladder:waterChanged</code>	<code>netId</code> , <code>src</code> , <code>ouvert</code>	Jet ouvert / fermé

Événement	Paramètres	Quand
masterladder:waterImpact	netId, src, coords	Impact du jet, 2 fois par seconde

Brancher un script de feu

```
-- server.lua de votre script de feu
AddEventHandler('masterladder:waterImpact', function(netId, src, coords)
    -- éteindre / affaiblir vos feux autour de coords (rayon ~3 m)
    MonScriptDeFeu.DamageAt(coords, 3.0, 25.0, src)
end)
```

Événements client

Événement	Paramètres	Quand
masterladder:client:placeChanged	netId, place	Ma place change (nil, nil quand je descends)
masterladder:client:waterImpact	coords, rayon	Impact du jet près de moi (feux éteints localement)