

Exports et événements

Exports serveur

```
-- Un foyer → fireId, incidentId (ou nil, erreur)
local fireId, incId = exports.MasterFire:CreateFire(coords, 'liquid', {
    size = 20, preset = nil, label = 'Station-service', dispatch = true,
    difficulty = 1.0, spreadChance = 35, bleve = false, visual = nil,
})

-- Plusieurs foyers = une intervention (un appel, un rapport) → incidentId, fireIds
local incId, fireIds = exports.MasterFire:CreateIncident({ c1, c2, c3 }, nil, { preset =
'house', dispatch = true })

exports.MasterFire:RemoveIncident(incId)
exports.MasterFire:RemoveFire(fireId)
exports.MasterFire:RemoveAllFires()

exports.MasterFire:GetIncidents()           -- interventions en cours (résumé)
exports.MasterFire:GetFire(fireId)         -- foyer : flammes, phase, chaleur, intérieur,
BLEVE...
exports.MasterFire:GetFires()
exports.MasterFire:GetNearestFire(coords, radius) -- → foyer, distance
exports.MasterFire:IsOnFire(coords, radius)
exports.MasterFire:IsLicensed()

-- Appel de confiance (canon, échelle, bras élévateur) : pas de contrôle de portée
local flammes = exports.MasterFire:ApplyAgent('cannon', impact, 2.5, 1.0, sourceJoueur)

exports.MasterFire:StartScenario(scenarioId, dispatch) -- → incidentId
exports.MasterFire:SetSpawner(true)
```

Erreurs possibles : `license` (licence non validée), `limit` (trop de feux), `origins` (foyers invalides), `type`, `coords`.

Exports client

```
exports.MasterFire:SprayAt('water', impact, 1.5)           -- validé par le serveur
exports.MasterFire:SetBreathingApparatus(true [, silencieux])
exports.MasterFire:HasBreathingApparatus()
exports.MasterFire:GetNearestFlame(coords, radius)         -- → fireId, flameId, distance
exports.MasterFire:IsNearFire(coords, radius)
exports.MasterFire:GetFires()
```

Événements serveur (AddEventHandler)

Événement	Paramètres
masterfire:server:incidentStarted	incidentId, { label, type, coords, fires, scenarioId }
masterfire:server:incidentClosed	incidentId, résumé (outcome extinguished / burned_out / cancelled, durée, intervenants, agents, phénomènes...)
masterfire:server:fireCreated	fireId, { type, coords, size, scenarioId, incidentId }
masterfire:server:fireRemoved	fireId, raison, { scenarioId, incidentId }
masterfire:server:phaseChanged	fireId, nouvellePhase, anciennePhase
masterfire:server:phenomenon	fireId, 'flashover' 'backdraft' 'bleve'
masterfire:server:dispatch	{ incidentId, label, typeLabel, foyers, coords, jobs }
masterfire:server:licenseChanged	valide