



CORE RX
ZERO-CODE XR FRAMEWORK

CORE RX

DOCUMENTAZIONE TECNICA

Edizione italiana - sorgente r147

CORE RX - FRAMEWORK

DOCUMENTAZIONE TECNICA

Edizione italiana



Sorgente di riferimento: r147 (47 script) - Unity 6000.5.4f1
Agosto 2026 - MAD-VF - Fabio Cecchini - www.mad-vf.store

Colophon e avvertenze

Questo documento accompagna il sorgente r147 di CORE RX e ne è la mappa completa: ogni regola, ogni contratto di montaggio, ogni silenzio e ogni costante, con riscontro file:riga. Non contraddice la guida utente: la integra. La guida resta la sede dell'uso e delle manopole (il suo capitolo 16 è il catalogo dei campi a inspector, qui richiamato scheda per scheda); questa documentazione è la sede del comportamento del codice. I tooltip dei campi vivono nel capitolo 16 della guida utente, citati VERBATIM dal sorgente; qui ogni voce porta l'identificativo esatto dell'inventario r147. I riferimenti rNNN nel testo (r104-r147) sono le revisioni interne di sviluppo del framework, numeri di collaudo sul ferro: non sono versioni del prodotto.

Le distribuzioni aggiornate del framework possono spostare qualche numero di riga: la coppia file più identificativo di ogni voce resta il riferimento buono. L'assistenza sul framework è inclusa: scrivimi a store@mad-vf.store dall'email del tuo account dello store (quello dell'acquisto, così ti riconosco subito) e rispondo a stretto giro. Scene su misura, prefab e modifiche custom sono a preventivo: stessa email con l'esigenza, e ricevi a stretto giro una stima di tempi e costi, con lavoro a consegna verificabile. MAD-VF di Fabio Cecchini - www.mad-vf.store

INDICE

PARTE I - FONDAZIONI	6
CAPITOLO 1 - ARCHITETTURA E FLUSSO DELLE SCENE	7
CAPITOLO 2 - CONTRATTI DI MONTAGGIO	8
CAPITOLO 3 - CONVENZIONI DI SORGENTE E DATI	9
PARTE II - SISTEMI	10
CAPITOLO 4 - BOOT, SCENE E RITORNI	11
CAPITOLO 5 - DISPLAY: LE QUATTRO MODALITÀ	12
CAPITOLO 6 - GIOCATORE: LOCOMOZIONE, AVATAR, DITA, SGUARDO	13
CAPITOLO 7 - FBT VIVE E STEAMVR	14
CAPITOLO 8 - FBT PICO	15
CAPITOLO 9 - COMBATTIMENTO E MAGIA	16
CAPITOLO 10 - ARMI E PRESA	17
CAPITOLO 11 - UI, HUD, AUDIO E FEEDBACK	18
CAPITOLO 12 - SPECCHI	19
CAPITOLO 13 - REGIA, ONDATE, DATI E RETE	20
PARTE III - RIFERIMENTO A SORGENTE	21
CAPITOLO 14 - LE 47 SCHEDE FILE	22
14.01 - RX_Abyss.cs	22
14.02 - RX_AudioEnding.cs	22
14.03 - RX_AvatarBuilder.cs	23
14.04 - RX_Blink.cs	26
14.05 - RX_BootSettings.cs	26
14.06 - RX_Character.cs	27
14.07 - RX_ComboManager.cs	28
14.08 - RX_DataBridge.cs	29
14.09 - RX_DisplayApplier.cs	30
14.10 - RX_EnemyAI.cs	31
14.11 - RX_FingerPoser.cs	35
14.12 - RX_GameDirector.cs	36
14.13 - RX_Haptics.cs	37
14.14 - RX_HitZone.cs	38
14.15 - RX_HowToPlay.cs	38
14.16 - RX_HowToPlayLauncher.cs	39
14.17 - RX_LaserGate.cs	39
14.18 - RX_LevelMusic.cs	40
14.19 - RX_MagicCaster.cs	40
14.20 - RX_MagneticGrab.cs	42
14.21 - RX_MirrorClone.cs	43
14.22 - RX_MirrorHandle.cs	44
14.23 - RX_Persistent.cs	44
14.24 - RX_PicoFBT.cs	45
14.25 - RX_PicoTrackerAdapter.cs	46
14.26 - RX_PlayerLocomotion.cs	47
14.27 - RX_RemoteLog.cs	50
14.28 - RX_SafeSpawner.cs	50
14.29 - RX_SecondWind.cs	51
14.30 - RX_Settings.cs	51
14.31 - RX_SimpleMirror.cs	52

14.32 - RX_UIGate.cs	52
14.33 - RX_UIManager.cs	52
14.34 - RX_UniversalProjectile.cs	53
14.35 - RX_VideoWipe.cs	54
14.36 - RX_ViveFBT.cs	55
14.37 - RX_ViveTrackerProfile.cs	57
14.38 - RX_WaveDirector.cs	58
14.39 - RX_WeaponDamage.cs	59
14.40 - RX_WeaponDropper.cs	60
14.41 - RX_WeaponRack.cs	61
14.42 - RX_WristHUD.cs	61
14.43 - MV_AlignRigToScene.cs	61
14.44 - MV_FrontOfCamera.cs	62
14.45 - MV_LoadSceneWithDelay.cs	62
14.46 - MV_SkipScene.cs	62
14.47 - MV_TrackerCube.cs	62
APPENDICI	64
APPENDICE A - LE 33 FRASI DI HOWTOPLAY (ITALIANO E INGLESE)	65
APPENDICE B - LA TAVOLA DEI SILENZI	68
APPENDICE C - LE COSTANTI	69

CAPITOLO 7

FBT VIVE E STEAMVR

Il motore è `RX_ViveFBT` (scheda 14.36): porta le caviglie tracciate sui bersagli IK del rig gambe con lisciatura esponenziale, guardia teletrasporto, offset radiale lungo l'asse anca-caviglia, boost del calcio, assistenza e tetto di estensione, e il piede a martello su terna ortonormale con isteresi anti flip. È anche la console della locomotion: `fbtMoveThreshold`, `trackerFrozenTimeout` e `hookCenterMaxRadius` vivono qui e vengono spinti sulla locomotion ogni frame (la locomotion è montata a runtime e non ha inspector persistente: la casa editabile è questa). PROCEDURA COMPLETA DELLE AZIONI `IsTracked` (il debito lasciato dal sigillo della guida, saldato qui).

Passo 1, la feature: abilitare `RX_ViveTrackerProfile` tra le feature `OpenXR Standalone` (categoria `Interaction`). La feature vive sull'estensione `XR_HTCX_vive_tracker_interaction`: senza estensione attiva nel runtime non si crea (`RX_ViveTrackerProfile.cs:56`), e il runtime di riferimento è `SteamVR`.

Passo 2, il layout: la feature registra il dispositivo `RX_ViveTracker` con matcher su interfaccia `XRInput` e prodotto `VIVE Tracker` (`RX_ViveTrackerProfile.cs:65`). Il layout espone il controllo `devicePose` (`usage Device`, `noisy`) più i controlli standard di ogni dispositivo `OpenXR`, tra cui `isTracked` AL LIVELLO ALTO del dispositivo.

Passo 3, i ruoli: due dispositivi logici, `left_foot` con caratteristica `Left` e `right_foot` con `Right` (`RX_ViveTrackerProfile.cs:87`): è la caratteristica a distinguere i lati nelle `binding mask`.

Passo 4, le azioni di posa: le azioni posizione dei tracker della locomotion (`leftTrackerPosAct` e `rightTrackerPosAct`) si legano a `devicePose/position` del dispositivo del proprio lato; la presenza si misura dai controls risolti (`RX_PlayerLocomotion.cs:138`).

Passo 5, le azioni `IsTracked`: due azioni dedicate, una per lato, legate al controllo `isTracked` DI LIVELLO ALTO del dispositivo (NON al figlio dentro il gruppo `devicePose`: quel figlio si è dimostrato morto, `r105`), e cablate in `tLeftIsTrackedAct` e `tRightIsTrackedAct` di `RX_ViveFBT` (`RX_ViveFBT.cs:26-27`).

Passo 6, il comportamento: entrambe cablate = autorità di certificazione: 2 su 2 a valore 1 e l'aggancio è istantaneo e certificato, senza rito del passo; qualunque altro valore = niente FBT, per progetto (il fallback del passo resta spento). Una sola cablata = autorità spenta per sicurezza, e anche qui il ripiego del passo torna attivo (canone `r128`: un solo lato = avviso e ripiego). Nessuna cablata = si torna all'handshake del passo: entrambi i tracker mossi oltre soglia, distanti tra loro, in quota plausibile (`RX_ViveFBT.cs:201-222`, `RX_PlayerLocomotion.cs:167`).

Passo 7, lo sgancio: sempre e comunque via sorveglianza: presenza dai controls e VITA del congelamento (`trackerFrozenTimeout`, misura in spazio locale del parent: il movimento del rig non maschera un tracker morto, `RX_PlayerLocomotion.cs:138-147`).

Passo 8, la verifica sul campo: `MV_TrackerCube` col suo semaforo (verde tracciato, rosso non tracciato, grigio binding non risolto) e il log `MV_CUBO` (scheda 14.47). Gli scenari dongle e tracker sono stati provati sul campo su Vive AIO il 2 agosto (0/0/1) e riverificati su SteamVR al passaggio platform 0, come registra il tooltip del campo (`RX_ViveFBT.cs:26`), citato per intero nel capitolo 16 della guida.

CAPITOLO 9

COMBATTIMENTO E MAGIA

Le regole del colpo, quelle che il giocatore vive, restano nella 6.5 della guida utente: quella è la sede, qui c'è il motore che le fa rispettare. Il giudice di ogni contatto è `RX_WeaponDamage` (scheda 14.39), montato su tutto ciò che ferisce: pugni e calci del player (fabbricati dal builder), hitbox dei nemici (fabbricate dall'AI), armi.

Misura la velocità vera del gesto in tempo non scalato, pretende il riarmo tra un colpo e l'altro (ritorno e nuova andata: niente danno per sfregamento; i calci a danno fisso del Second Wind ne fanno a meno, righe 112-113 e 121), applica il cooldown per bersaglio, il dimezzamento del pugno a mano aperta, i lati (l'arma di nessuno o in mano al player non colpisce mai il player, righe 128-129; quella del nemico armed invece lo ferisce: righe 119-154), la finestra melee del nemico (un attacco, un colpo) e l'usura delle armi a usi finiti. Le zone del corpo sono targhette `RX_HitZone` (scheda 14.14).

Il player ne monta dodici (righe 193-205 del builder): danno sulla testa, sul tronco e sulle cosce; parata su avambracci e stinchi, che fermano melee e magia; e mani e piedi, che fermano la sola magia (righe 202-205). Il nemico di terra ne monta otto (`RX_EnemyAI.cs:96-104`): danno su testa, tronco e cosce e parata su avambracci e stinchi, senza mani e piedi paranti. Il volante non ne monta nessuna (ramo della riga 57): prende danno su tutto il corpo. Il corpo con `usesHitZones` fuori zona non prende nulla: si colpisce dove il corpo dichiara di poter essere colpito.

Il cervello avversario è `RX_EnemyAI` (scheda 14.10): due nature (terra e cielo), fasce di distanza (melee entro `attackRange`, magia nella fascia `castMin-castMax`, inseguimento oltre), token di attacco dalla regia, ritirate al 50 per cento dopo il colpo, kamikaze col telegraph dato dalla distanza minima di picchiata (il volante armato la ignora: la picchiata è la sua unica mossa, riga 248). La magia vola su `RX_UniversalProjectile` (scheda 14.34), uguale per tutti: homing con finestra di schivata, annullamento magia contro magia, danno a velocità per la magia del player e BLAST ravvicinato a danno doppio con tutte le ricompense.

Le combo sono automi alla Knuth-Morris-Pratt su `RX_ComboManager` (scheda 14.07): sequenze a inspector, mosse generiche che sussumono le laterali, premi in danno, punti, mana e cura. Il Second Wind (scheda 14.29) è l'ultima parola: slow motion in tempo reale, un colpo a segno per rialzarsi, una sola volta a partita. La magia del player parte dalla mano: `RX_MagicCaster` (scheda 14.19) governa il mana con le politiche di scena (pieno nei menu, zero nel finale), pretende mano libera e niente UI sotto tiro, spende in anticipo e rimborsa solo nei casi giusti (pausa, pugno entro la grazia), e al rilascio la scaglia lungo l'asse del braccio: si punta col braccio, che il controller conduce. Il canone è sigillato: laser libero e mano libera.

CAPITOLO 13

REGIA, ONDATE, DATI E RETE

La regia di partita è doppia. `RX_GameDirector` (scheda 14.12) governa i passaggi di stato: avvio dell'arena, pausa col congelamento coerente (`timeScale` a zero col valore precedente restituito esatto: gli stati in *slow motion* riprendono da dove erano), finale con ripristino completo, e uscita dal gioco. `RX_WaveDirector` (scheda 14.38) governa il ritmo: tempo di partita, corsie di spawn sotto i massimi ogni 5 secondi, token di attacco a giri con mescolata Fisher-Yates, ricalcolo immediato a ogni morte e ritento breve quando tutti battono in ritirata.

Il fattore di difficoltà è una formula sola (`DifficultyFactor`: difficoltà e `arenaLevel`, minimo 0.1): moltiplica la salute dei nemici e il danno subito dal player (`RX_Character.cs:75` e 121); le cure di combo e blast seguono la difficoltà per altra via, a unità fisse sull'anagrafe. L'ostetrico è `RX_SafeSpawner` (scheda 14.28): posa a terra campionata sul `NavMesh` lontano dal player, posa in aria sull'anello verso il bordo, e tre ingressi coreografati (portale, emersione, risalita) col nemico congelato. Il becchino è `RX_Abyss` (scheda 14.01): soccorso per il player fin sopra la soglia di vita minima (a soglia o sotto, fine partita), inceneritore per tutto il resto.

I dati sono tre archivi JSON parsati da `RX_DataBridge` (scheda 14.08): array nudi, involucro a runtime, ultima voce a parità di ID, ripieghi globali per clip e frasi dei nemici. I file di riferimento vivono in `JSON_build` (`players`, `weapons`, `enemies`) e rispettano le forme dei tre profili; portano anche chiavi predisposte per le edizioni future (il blocco `ai_pattern` dei nemici, `weakness`, `staggerImpact` e simili) che questo motore non legge: compilarle non cambia nulla (nota di onestà della guida, appendice E).

La rete è volutamente minima: `RX_RemoteLog` (scheda 14.27) cattura i log e li spedisce come POST di testo grezzo a un endpoint PHP; lo script `Server_debug/log.php` accetta il corpo fino a 5 MB e lo scrive in `log.txt` accanto a sé, rispondendo ok (`log.php:2-7`). Niente account, niente telemetria: un canale di debug a comando, e il quit è garantito anche a server morto.

CAPITOLO 14

LE 47 SCHEDE FILE

(per ogni file: ruolo, il conto dei campi con rimando alla guida cap. 16 e, prima di ogni altra voce, gli eventuali campi nascosti; poi REGOLE, CONTRATTI, VINCOLI DI PIATTAFORMA, SILENZI e COSTANTI in un unico filo, nell'ordine del sorgente, ogni voce con riscontro file:riga. L'unità di conto del censimento è la voce, 461 per i campi: 431 voci tra i campi a inspector, che sono 433 campi con due coppie appaiate alla 14.26; e 30 voci tra i campi nascosti, che sono 40 campi con dieci coppie appaiate, cinque alla 14.24 e cinque alla 14.36, sempre stampate con entrambi i nomi)

14.01 - SCHEDA RX_Abyss.cs

Ruolo: sensore di caduta dell'arena. È un volume trigger da montare sotto il pavimento: tutto ciò che vi entra viene smistato per natura (player, nemico, cadavere, magia, arma) e trattato di conseguenza. Per il player è un istituto di soccorso, per tutto il resto è un inceneritore.

Campi: 4 campi a inspector, tutti trattati nel capitolo 16 della guida utente (rimando). Nessun campo nascosto.

CONTRATTO (RX_Abyss.cs:13) **RX_Character sul parent:** il riconoscimento usa GetComponentInParent di RX_Character: player e nemici devono esporre il componente nella gerarchia sopra il collider che tocca il trigger.

REGOLA (RX_Abyss.cs:16) **debounce cadute player:** una caduta entro 1 secondo dalla precedente viene ignorata con return silenzioso; la costante 1f è cablata nel sorgente.

REGOLA (RX_Abyss.cs:18) **gate soccorso:** con CurrentHp sopra rescueMinHpUnits scattano penalità (ApplyFallPenalty) e teleport; altrimenti la partita finisce direttamente. Second Wind non viene mai innescato dall'abisso.

CONTRATTO (RX_Abyss.cs:20) **RX_PlayerLocomotion.TeleportRig:** il teleport è delegato alla locomotion trovata con FindFirstObjectByType; destinazione playerRespawn, o l'origine del mondo se il campo è vuoto.

SILENZIO (RX_Abyss.cs:22) **locomotion assente:** senza RX_PlayerLocomotion in scena la penalità si applica ma il teleport non avviene, senza segnale.

CONTRATTO (RX_Abyss.cs:24) **RX_GameDirector.GoToEndingScene:** la fine partita da abisso passa dal GameDirector, se presente in scena.

REGOLA (RX_Abyss.cs:28) **cadaveri:** un nemico già morto che cade viene solo distrutto: niente punti, niente conteggi di ondata.

REGOLA (RX_Abyss.cs:31) **nemici volanti:** il bonus groundFallBonus premia solo la caduta dei nemici non volanti; il volante caduto non vale punti.

CONTRATTO (RX_Abyss.cs:31) **RX_WaveDirector.Instance:** punteggio e rimozione dal conteggio ondata (RemoveEnemy col flag volante) passano dal WaveDirector, se presente.

REGOLA (RX_Abyss.cs:33) **pulizia oggetti:** ciò che entra con tag Magic o Weapon viene distrutto: l'abisso è anche il cestino delle armi perse.

14.02 - SCHEDA RX_AudioEnding.cs

Ruolo: regista del finale solo audio. Riporta la camera allo skybox, suona l'eventuale frase d'ingresso e poi la clip di finale, e torna al Boot in ogni caso: a fine clip, a tasto premuto o in mancanza della fonte. Ogni attesa corre in tempo reale, immune alla scala del tempo.

APPENDICE B

LA TAVOLA DEI SILENZI

Sedici punti in cui il framework fallisce senza dire nulla: è una scelta (il frame non si sacrifica mai a un log), ma va conosciuta. Per ogni silenzio: dove, cosa tace, come accorgersene. Il dettaglio completo sta nella scheda del file.

1. RX_Abyss.cs:22, locomotion assente: la penalità di caduta si applica ma il teleport non avviene. Sintomo: il player ferito resta nel vuoto.
2. RX_AudioEnding.cs:64, camera assente: il riposizionamento del canvas LOADING salta, senza segnale. Sintomo: nessuno visibile nel finale audio, dove il canvas vive acceso solo dentro il blocco sincrono (righe 41-45) e non arriva a schermo; la scritta ferma nel nulla si vede invece nel gemello video, che tiene il canvas acceso durante la preparazione (RX_VideoWipe.cs:50-74).
3. RX_AvatarBuilder.cs:98, catch vuoto: ogni eccezione del montaggio automatico tracker viene inghiottita. Sintomo: riferimenti tracker mancanti senza errori in console.
4. RX_AvatarBuilder.cs:123, ossa braccia mancanti: il rig braccia non nasce. Sintomo: braccia dell'avatar che non seguono le mani.
5. RX_Blink.cs:32, camera assente: niente overlay e niente battiti. Sintomo: risveglio senza palpebra.
6. RX_DisplayApplier.cs:72, arenaFloor vuoto: nessun anello di confine. Sintomo: XR o AR senza anello sul perimetro.
7. RX_DisplayApplier.cs:78, bounds introvabili: senza Renderer né Collider sul pavimento, nessun anello. Sintomo: come sopra.
8. RX_DisplayApplier.cs:86, raggio non positivo: margine più largo del pavimento, nessun anello. Sintomo: come sopra, con pavimenti piccoli.
9. RX_EnemyAI.cs:66, nemico a terra non humanoid: niente hitbox, zone e FirePoint automatici. Sintomo: nemico che non colpisce e non para.
10. RX_Haptics.cs:40, catch vuoto: ogni eccezione aptica inghiottita. Sintomo: niente vibrazioni, nessun errore.
11. RX_HitZone.cs:12, osso nullo: Build restituisce null. Sintomo: zona mancante su un rig senza l'osso richiesto.
12. RX_HowToPlay.cs:85, operazione nulla: il ritorno al Boot non avviene (scena fuori dai Build Settings). Sintomo: tutorial che non esce.
13. RX_MirrorClone.cs:17, dipendenze assenti: nessun clone allo specchio. Sintomo: specchio che inquadra tutta la scena, avatar vero compreso: la culling mask resta ai default, la stretta della riga 60 non arriva mai.
14. RX_SafeSpawner.cs:19, pool vuoto: nessuno spawn dalla corsia. Sintomo: arena senza nemici di terra o di aria.
15. RX_SafeSpawner.cs:35, tentativi esauriti: 15 pose rifiutate, nessuno spawn. Sintomo: spawn rari con NavMesh stretti o player al centro.
16. RX_SimpleMirror.cs:15, renderer assente: lo specchio non si costruisce. Sintomo: schermo dello specchio inerte.

FINE DELL'ANTEPRIMA

Questa anteprima gratuita contiene 11 pagine riprese pari pari dal documento ufficiale: la copertina, il frontespizio, il colophon coi contatti in prima pagina, l'indice completo, tre capitoli di sistema (FBT VIVE e SteamVR; Combattimento e magia; Regia, ondate, dati e rete), la prima pagina delle 47 schede file e la tavola dei silenzi, oltre al retro di copertina qui in chiusura.

LA DOCUMENTAZIONE TECNICA COMPLETA

CORE RX - Framework - Documentazione Tecnica, edizione italiana: **72 pagine A4**. Il volume di ingegneria che affianca la Guida utente: tredici capitoli su architettura e sistemi, poi le 47 schede file - una per script del sorgente, con il ruolo, i campi e ogni regola, contratto, vincolo di piattaforma, silenzio e costante - e tre appendici. **1202 voci di inventario**, ciascuna con il suo riferimento esatto file:riga al sorgente r147: 461 manopole, 558 regole, 146 contratti, 12 vincoli di piattaforma, 16 silenzi, 9 costanti. Cosa fa il codice, dove lo fa e cosa tace di proposito: verificato enunciato per enunciato sul sorgente, pagina per pagina.

NELLA CONFEZIONE

- Guida utente completa in italiano e in inglese (84 pagine ciascuna)
- Documentazione Tecnica in italiano e in inglese (72 pagine ciascuna)
- EULA in italiano e in inglese
- Quick Reference in italiano e in inglese (4 pagine, con la checklist per il cambio di piattaforma)
- Il sorgente del framework: 42 script + 5 tool, con inventario del sorgente a 1202 voci

LICENZA E ASSISTENZA

Gli aggiornamenti del framework sono **INCLUSI** nella licenza (EULA, sez. 6) e distribuiti senza costi aggiuntivi. L'assistenza sul framework è inclusa: scrivi dall'email del tuo account sullo store e ricevi riscontro a stretto giro. Scene su misura, prefab e modifiche custom sono a preventivo: stima di tempi e costi a stretto giro, si lavora a consegna verificabile.

PROVALA PRIMA, GRATIS

FFRU DEMO RC è creata con CORE RX come suo engine: scaricala gratis da www.fightforresurrection.app/crx e gioca, per vedere con mano cosa puoi creare.

mad-vf.store

MAD-VF - Fabio Cecchini

Meta Quest - PICO - VIVE - SteamVR

CORE RX

DOCUMENTAZIONE TECNICA

La mappa completa di ciò che il framework fa sotto il cofano.

Il sorgente r147 è la fonte di verità: 47 script letti riga per riga; ogni regola, ogni contratto di montaggio, ogni silenzio documentati con riscontro file:riga.

Le quattro modalità - VR, MR, XR, AR - spiegate dal codice che le governa.

I motori full-body: PICO, VIVE, SteamVR - e Meta Quest via Link (platform 0, tracker VIVE).

461 manopole, 558 regole, 146 contratti, 12 vincoli di piattaforma, 16 silenzi, 9 costanti: censiti, uno per uno.

Nessuna scatola nera: quello che leggi qui è quello che gira.

Assistenza inclusa: scrivimi a store@mad-vf.store dall'email del tuo account (quello dell'acquisto) - rispondo a stretto giro. Personalizzazioni a preventivo: stessa email, stima di tempi e costi a stretto giro, consegna verificabile.

MAD-VF - www.mad-vf.store