Keskustellaan rempy pelin valikosta jossa päähenkilö voi valita 10 eri työkohteen väliltä joka aamu töihin alkaessaan. Kukin työ etenee laskurilla vaihe vaiheelta suoraviivaisesti kohti loppuaan. Työn saattaa saada tehtyä täydellisesti, enemmän tai vähemmän hyvin tai se voi mennä pieleen.

Tämä kuulostaa mielenkiintoiselta ja toimivalta pelimekaniikata! Se muistuttaa hieman simulaatio- ja resurssienhallinta pelejä, joissa päivittäiset valinnat vaikuttavat hahmon kehitykseen ja tarinan etenemiseen.

Jotta tästä valikosta ja työskentelyprosessista tulisi mahdollisimman koukuttava, tässä on muutamia ajatuksia ja ehdotuksia, miten mekaniikkaa voisi syventää:

---

1. Valikkorakenne: "Aamun kymmenen valintaa"

Jotta 10 vaihtoehtoa ei tunnu pelaajasta uuvuttavalta, valikon tulisi tarjota selkeää informaatiota yhdellä silmäyksellä. Jokaisessa "työkontissa" voisi olla:

- **Vaikeustaso:** (Esim. 1–5 tähteä). Vaikuttaa siihen, kuinka helposti työ menee pieleen.
- **Palkkio:** Paljonko rahaa tai mainetta työstä saa onnistuessaan.
- **Kesto:** Kuinka monta "askelta" laskurissa on.
- **Riskikerroin:** Mahdollisuus kriittiseen epäonnistumiseen vs. jättipottiin.

2. Työn eteneminen ja laskurimekaniikka

Vaikka eteneminen on suoraviivaista, pelaajalle voi antaa hallinnan tunnetta "laskurin rullatessa":

- **Väliinteraktiot:** Kun laskuri saavuttaa esimerkiksi 25 %, 50 % ja 75 % kohdan, pelaajalle voi tulla pieni valintatilanne (esim. "Kone ylikuumenee: a) Hidasta tahtia, b) Jatka riskillä").
- **Hahmon taidot:** Pelaajan statsit (esim. keskittymiskyky, voima tai kokemus) voivat vaikuttaa siihen, kuinka nopeasti laskuri liikkuu tai kuinka leveä "onnistumisalue" on.

3. Tulostaso ja seuraukset

On hienoa, että lopputulos ei ole vain onnistuminen tai epäonnistuminen. Tulosasteikko voisi näyttää tältä:

Tulos	Kuvaus	Vaikutus

Täydellinen	Työ tehty virheettömästi ja etuajassa.	Bonusraha, maineen kasvu, stressin lasku.
Hyvä	Standardisuoritus.	Normaali palkka.
Välttävä	Työ meni läpi, mutta hutiloiden.	Pieni palkka, työnantajan tyytymättömyys.
Pieleen	Työ epäonnistui tai materiaalit menivät hukkaan.	Ei palkkaa, korvausvelvollisuus, stressin nousu.

4. Visuaalinen ja kerronnalinen toteutus

Koska kyseessä on "rempy" (viitaten kenties remontointiin tai rakentamiseen?), työkontit voisivat olla konkreettisia työmaita kartalla tai listalla.

- **Aamun fiilis:** Päähenkilön kunto (nälkä, väsymys) voisi rajata osan konteista pois tai tehdä vaikeista töistä entistä riskialttiimpia.
- **Erikoistuminen:** Jos pelaaja tekee jatkuvasti tietyntyyppisiä kontteja (esim. sähkötöitä), hänestä tulee niissä mestari, jolloin "Täydellinen" tulos on helpompi saavuttaa.

Miten tämä vaikuttaa pelin kulkuun?

Onko tarkoituksena, että pelaaja yrittää säästää rahaa johonkin tiettyyn tavoitteeseen, vai onko työn laadulla suurempi merkitys tarinan juonenkäänteisiin (esim. huonosti tehty työ aiheuttaa ongelmia myöhemmin)?

Haluaisitko, että hahmottelen tarkemmin esimerkiksi noita "työn aikana tapahtuvia valintoja" tai sitä, miltä tuo kymmenen listan visuaalinen näkymä voisi näyttää?

Aikalailla näin. Hahmon taidot ja tila sekä häiriöt päivän aikana vaikuttavat. Tarkoitus on korjata moottoripyörä romusta ajokuntoiseksi ja tehdä tarvittavat säädöt. Tavoite on että pyörä kestää hajoamatta ja ylikuumentumatta ajoa koko päivän mittaisen matkan.

Tämä selkeyttää pelin tavoitetta huomattavasti! Kyseessä ei olekaan vain "työ", vaan projektinhallinta, jossa jokainen valinta vaikuttaa lopputulokseen: kestääkö pyörä suuren matkan vai jättääkö se tielle.

Koska tavoitteena on moottoripyörän kunnostus ja säätäminen, tässä on ehdotus siitä, miten nuo **10 työkonttia** voisivat jakautua ja miten mekaniikka toimii suhteessa lopulliseen tavoitteeseen.

---

1. Moottoripyörän 10 osa-aluetta (Työkontit)

Pelaaja valitsee aamulla, minkä osa-alueen kimppuun hän käy. Jokainen kontti vaatii useita työvaiheita (laskuri).

1. **Moottorin alakerta:** Kiertokanget, laakerit (Kriittinen kestävyyden kannalta).
2. **Sylinteri ja mäntä:** Puristukset ja teho.
3. **Kaasutin/Ruisku:** Polttoaineen syöttö ja seokset (Vaikuttaa ylikuumenemiseen).
4. **Jäähdytysjärjestelmä:** Syyläri, letkut ja nesteet (Kriittinen ylikuumenemisen kannalta).
5. **Sähköt ja sytytys:** Tulpat, johdot, akku.
6. **Voimansiirto:** Kytkin, vaihteisto ja ketjut.
7. **Jarrut:** Turvallisuus ja hallinta.
8. **Jousitus ja runko:** Ajomukavuus ja vakaus.
9. **Renkaat ja vanteet:** Pito ja tasapaino.
10. **Viimeistely ja koekäyttö:** Pulttien kiristys, hienosäätö ja testaus.

---

2. Päivän kulku ja häiriötekijät

Kun pelaaja aloittaa työn (laskuri alkaa rullata), peli simuloi päivän tapahtumia:

- **Hahmon tila:** Jos hahmo on väsynyt, laskuri etenee hitaammin ja "virhealue" kasvaa.
- **Satunnaiset häiriöt:**
    - *Naapuri tulee juttelemaan:* Keskeyttää työn, laskuri pysähtyy. Jos pelaaja on epäkohtelias, stressi nousee. Jos kohtelias, aikaa kuluu hukkaan.
    - *Työkalu katoaa/rikkoutuu:* Pelaajan on käytettävä aikaa sen etsimiseen tai korjaamiseen.
    - *Puuttuva osa:* Jos pelaaja huomaa kesken työn, että osa puuttuu, työn laatu kärsii ("purkkaviritys"), ellei hän tilaa uutta ja odota päivää.

---

3. Työn tuloksen vaikutus "Matkamittariin"

Kun työ on tehty, se tallentuu pyörän tilaan. Tämä on tärkeää "suuren matkan" kannalta:

Työn laatu	Vaikutus pyörään	Esimerkki (Jäähdytys

Täydellinen	Osa on parempi kuin uusi.	Jäähdytys toimii optimaalisesti myös helteellä.
Hyvä	Toimii kuten pitääkin.	Normaali lämmönhallinta.
Välttävä	Pieni riski pettämiseen.	Neste saattaa tihkua, ylikuumenemisriski nousee pitkässä ajossa.
Pieleen	Piilevä vika.	Jäähdytin saattaa haljeta tai letku irrota matkan puolivälissä.

---

4. Loppuhuipentuma: Matka

Kun pyörä on "valmis", alkaa lopputesti. Tämä voisi olla visualisoitu matka, jossa peli käy läpi eri osa-alueet:

- **Moottoritie-osio:** Testaa moottorin kestävyyttä (alakerta).
- **Vuoristo/Ruuhka:** Testaa jäähdytystä (ylikuumeneminen).
- **Mutkainen tie:** Testaa jousitusta ja renkaita.

Jos jokin osa-alue tehtiin "välttävästi", pelaaja joutuu matkan aikana tekemään **pikakorjauksia tien päällä**, mikä kuluttaa aikaa ja resursseja. Jos se tehtiin "päin mäntyä", pyörä hajoaa ja peli päättyy keskeytykseen.

Miten Renpy-toteutus kannattaisi tehdä?

Renpylla tämä onnistuu parhaiten käyttämällä `screen`-työkaluja valikoihin ja `variables`-muuttujia seuraamaan kunkin 10 kontin kuntoa (esim. `engine_quality = 0-100`).

Haluaisitko nähdä esimerkin siitä, miltä "työskentely-screen" ja sen laskuri voisi kooditasolla (tai logiikaltaan) näyttää, jotta se tuntuu dynaamiselta?

No juuri tätä on tarkoitus, avaa renpy logiikkaa miten tää saadaan tehtyä dynaamisesti eikä monotonisesti

Jotta pelimekaniikasta saadaan dynaaminen ja "elävä", meidän on siirryttävä pois pelkästä tekstin klikkailusta ja hyödynnettävä Ren'Py:n Python-integraatiota. Avain monotonisuuden välttämiseen on **vaihtelu, visuaalinen palaute ja yllätyksellisyys.**

Tässä on ehdotus logiikasta, joka on jaettu kolmeen osaan: **Data-rakenne**, **Työskentely-looppi** ja **Satunnaistapahtumat**.

---

1. Työkohteiden määrittely (Python-objektit)

Sen sijaan, että kirjoittaisit 10 erillistä koodinpätkää, määritellään työt Python-luokkana. Tämä tekee koodista helposti hallittavaa.

Python

```
init python:
    class TyoKohde:
        def __init__(self, nimi, vaikeus, kuvaus):
            self.nimi = nimi
            self.vaikeus = vaikeus  # 1-10
            self.kuvaus = kuvaus
            self.edistys = 0        # 0-100
            self.laatu = 100        # Alkaa sadasta, laskee virheistä
            self.valmis = False

    # Luodaan lista työkontteja
    tyolista = [
        TyoKohde("Moottorin alakerta", 8, "Kaiken perusta. Vaatii tarkkuutta."),
        TyoKohde("Kaasuttimen säätö", 5, "Hienovaraista työtä polttoaineen kanssa."),
        # ... loput 8 tähän
    ]
```

2. Dynaaminen Työskentely-looppi

Monotonisuus rikotaan sillä, että laskuri ei vain liiku, vaan se "elää" hahmon tilan mukaan. Käytetään Ren'Py:n `screen`-ominaisuutta näyttämään edistyminen reaaliajassa.

Logiikka työn edetessä:

1. **Nopeus:** Riippuu hahmon taidosta ja energiasta.
2. **Laatu:** Joka askeleella heitetään "noppaa". Jos hahmo on väsynyt tai taitotaso on matala, `laatu` laskee hieman.
3. **Keskeytykset:** Pieni mahdollisuus (esim. 5%), että laskuri pysähtyy ja tapahtuu häiriö.

3. Esimerkki: "Työpäivä"-koodi

Tämä on pelkistetty logiikka siitä, miten yksi työsuoritus voisi edetä dynaamisesti:

Koodinpätkä

```
label aloita_tyo(kohde):
    $ kohde.edistys = 0
    $ kohde.laatu = 100

    show screen tyo_palkki(kohde)  # Näytetään edistymispalkki ruudulla

    while kohde.edistys < 100:
        $ renpy.pause(0.2) # Laskuri liikkuu itsekseen

        # LASKENTA: Edistyminen hahmon kunnon mukaan
        $ nopea_lisays = (pelaaja.taito * 0.5) + (pelaaja.energia * 0.2)
        $ kohde.edistys += nopea_lisays

        # LAADUN VALVONTA: Mitä väsyneempi, sitä enemmän virheitä
        if renpy.random.randint(1, 100) > pelaaja.keskittymiskyky:
            $ kohde.laatu -= 1
            "*(Kiroilua, jokin osa ei istu heti paikalleen...)*"

        # SATUNNAISET HÄIRIÖT
        if renpy.random.randint(1, 100) < 5: # 5% mahdollisuus keskeytykseen
            call hairio_tapahtuma
            if _return == "keskeytä":
                hide screen tyo_palkki
                return "keskeytetty"

    hide screen tyo_palkki
    call tyon_arviointi(kohde)
    return
```

---

4. Miten vältetään monotonisuus? (Käyttäjäkokemus)

- **Visuaalinen palaute:** Palkin väri voisi muuttua vihreästä keltaiseksi tai punaiseksi, jos laatu-muuttuja laskee liikaa. Tämä luo paniikkia ja jännitystä.
- **Äänimaailma:** Työkalujen äänet (hylsyn naksahdus, metallin kilinä) nopeutuvat tai hidastuvat laskurin tahdissa.
- **Tekstivariantit:** Sen sijaan, että lukisi aina "Teet töitä", peli valitsee listasta satunnaisia lauseita: *"Kiristät pultteja ranteet kipeinä..."*, *"Öljyä roiskuu hihallesi, mutta jatkat..."*.
- **Kriittiset hetket:** Kun laskuri on 90% kohdalla, peli voi pysähtyä ja kysyä: *"Haluatko tehdä viimeistelyn huolella (vie aikaa) vai hutaisten (riski)?"*

Työn lopputulos (Arviointi)

Lopussa katsotaan kohde.laatu muuttujaa:

- **95-100:** "Täydellinen" – Pyörä kehrää kuin kissa.
- **70-94:** "Hyvä" – Toimii, mutta ehkä hieman tärisee.
- **40-69:** "Välttävä" – Moottori käy kuumana, toivottavasti kestää.
- **Alle 40:** "Pieleen" – Osia jää yli, pyörä savuttaa jo tallissa.

Miltä tämä rakenteellinen lähestymistapa vaikuttaa? Haluaisitko, että syvennymme tarkemmin tuohon "hairio_tapahtuma" -logiikkaan tai siihen, miten kymmenen työn valikko dynaamisesti päivittyy aamuisin?

Katsotaan molempia, otetaan häirio_tapahtuma ensin.

Toteutetaan ensin **häiriölogiikka**, sillä se on avainasemassa rikkomaan monotonisuutta. Sen jälkeen katsotaan, miten **valikko** mukautuu päivän mittaan.

1. Häiriötapahtumat (`hairio_tapahtuma`)

Häiriön ei pitäisi olla vain "paina nappia" -ilmoitus, vaan pieni taktinen valinta. Luodaan pooli erilaisia häiriöitä, jotka vaikuttavat eri muuttujiin (aika, laatu, hahmon kunto).

Koodiesimerkki häiriölogiikasta:

Python

```
init python:
    import random

    # Määritellään häiriöt sanakirjana
    hairiot = [
        {
            "id": "naapuri",
            "otsikko": "Utelias naapuri",
            "teksti": "Naapurin Reiska tulee aidan yli kyselemään moottorin ajoituksista. Hänellä riittää
juttua...",
            "valinnat": [
                ("Ole kohtelias ja kuuntele", {"aika": -30, "stressi": -5, "laatu": 0}),
                ("Sano, että on kiire", {"aika": -5, "stressi": 10, "laatu": -5})
            ]
        },
        {
            "id": "tyokalu_hukassa",
            "otsikko": "Työkalu kadoksissa",
            "teksti": "Missä se 10mm hylsy taas on?! Se oli tässä juuri äsken!",
            "valinnat": [
                ("Etsi rauhassa", {"aika": -15, "stressi": 2, "laatu": 5}),
                ("Käytä jakoavainta (riski)", {"aika": -5, "stressi": 5, "laatu": -15})
            ]
        },
        {
            "id": "nalka",
            "otsikko": "Vatsa kurnii",
            "teksti": "Keskittymisesi herpaantuu nälän vuoksi. Kädet alkavat täristä.",
            "valinnat": [
                ("Pidä evästauko", {"aika": -20, "energia": 20, "laatu": 0}),
```

```
            ("Paina menemään väkisin", {"aika": 0, "energia": -10, "laatu": -20})
        ]
    }
]
```

Ren'Py label häiriön käsittelyyn:

Koodinpätkä

```
label hairio_tapahtuma(nykyinen_kohde):
    $ h = random.choice(hairiot)
    $ otsikko = h["otsikko"]
    $ teksti = h["teksti"]

    "HÄIRIÖ: [otsikko]"
    "[teksti]"

    menu:
        "[h[valinnat][0][0]]":
            $ efektivaihtoehto = h["valinnat"][0][1]
        "[h[valinnat][1][0]]":
            $ efektivaihtoehto = h["valinnat"][1][1]

    # Sovelletaan vaikutukset
    $ kuluva_aika += efektivaihtoehto.get("aika", 0)
    $ pelaaja.stressi += efektivaihtoehto.get("stressi", 0)
    $ pelaaja.energia += efektivaihtoehto.get("energia", 0)
    $ nykyinen_kohde.laatu += efektivaihtoehto.get("laatu", 0)

    "Jatkat töitä, mutta tilanne muuttui."
    return
```

---

2. Dynaaminen Valikko (10 Työkonttia)

Valikon pitää kertoa pelaajalle yhdellä silmäyksellä, mitä on tehty ja mihin kannattaa keskittyä. Ren'Py `screen` on tähän paras työkalu.

Valikon ominaisuudet:

- **Värikoodaus:** Tehdyt työt vihreinä, tekemättömät harmaina.
- **Tila-ikonit:** Jos jokin osa on tehty "Välttävästi", se voisi näkyä keltaisena huutomerkkinä (pelaaja voi haluta korjata sen myöhemmin, jos aikaa jää).
- **Ajan hallinta:** Jokainen työ vie tietyn määrän tunteja. Jos ilta hämärtää, vaikeat työt muuttuvat riskialttiimmiksi.

Screen-koodin hahmotelma:

Koodinpätkä

```
screen tyovaihe_valikko():
    frame:
        xalign 0.5 yalign 0.5
        vbox:
            spacing 10
            text "Tallin työlista - Valitse päivän kohde" size 30

            for kohde in tyolista:
                hbox:
                    spacing 20
                    # Näytetään laatuindikaattori jos työ on tehty
                    if kohde.valmis:
                        text "[kohde.nimi]: VALMIS (Laatu: [kohde.laatu]%)" color "#00ff00"
                    else:
                        # Painike työn aloittamiseksi
                        textbutton "[kohde.nimi]":
                            action [SetVariable("nykyinen_tyo", kohde), Return("aloita")]
                            sensitive not kohde.valmis # Ei voi valita jos jo tehty

                        text "Vaikeus: [kohde.vaikeus]" size 14

            textbutton "Lopeta päivä / Testiajo":
                yoffset 20
                action Return("testiajo")
                background "#aa0000"
```

---

3. Logiikan yhdistäminen: "Päivittäinen kierto"

Tämä on se dynaaminen osuus: peli ei ole vain lista, vaan resurssien hallintaa.

1. **Aamu:** Hahmon tila arvotaan (onko herännyt hyvin vai huonosti?).
2. **Valikko:** Pelaaja valitsee yhden kymmenestä kontista.
3. **Työ-looppi:** Laskuri rullaa. Häiriöitä tapahtuu. Pelaaja tekee valintoja.
4. **Palaute:** Työ valmistuu. Laatu tallentuu (esim. `kaasutin_laatu = 85`).
5. **Iltapäivä:** Onko aikaa vielä toiseen työhön? Vai onko hahmo liian väsynyt?
6. **Yö:** Lepo palauttaa energiaa, mutta kestävyysmatka lähestyy.

Miten vältetään monotonisuus koodilla?

- **Laskurin nopeuden vaihtelu:** `kohde.edistys += random.uniform(0.5, 2.0) * pelaaja.vireystila`. Laskuri ei liiku tasaisesti, vaan "nykii" hahmon mukaan.
- **Kriittiset onnistumiset:** Jos työn aikana `laatu` pysyy yli 95:ssä pitkään, peli voi antaa "Flow-tilan", jolloin häiriöitä ei tapahdu ja edistys on nopeaa.
- **Visuaalinen kerronta:** Kun pelaaja korjaa esim. "Moottorin alakertaa", taustalla oleva kuva moottorista muuttuu: ruoste katoaa ja uusia osia ilmestyy (tämä onnistuu Ren'Py:ssä helposti `ConditionSwitchillä`).

Haluatko seuraavaksi katsoa, miten tuo lopullinen "kestävyysmatka" lasketaan näiden kymmenen osan laadun perusteella? Se on se kohta, missä pelaajan virheet alkavat kummitella.

Juu kerro "kestävyysmatkasta"

Kestävyysmatka on pelin "loppuvastus". Se on dynaaminen simulaatio, jossa aiemmin tekemäsi työ kohtaa todellisuuden. Se ei ole vain yksi animaatio, vaan sarja tarkistuspisteitä, joissa moottoripyörän eri osa-alueiden laatu joutuu koetukselle.

Tässä on logiikka, jolla matkasta saadaan jännittävä ja pelaajan valintoihin pohjautuva:

---

1. Matkan muuttujat (Hidden Stats)

Kun matka alkaa, peli alkaa seurata kolmea kriittistä arvoa, jotka elävät koko matkan ajan:

- **Lämpötila (`temp`):** Nousee, jos jäähdytys tai seokset on säädetty huonosti.
- **Värinä (`vibration`):** Kasvaa, jos alakerta tai runko on huterasti koottu.
- **Varmuus (`reliability`):** Yleisarvo, joka laskee jokaisen pienen "häiriön" myötä.

---

2. Matkan vaiheet ja "Noppaheitot"

Matka kannattaa jakaa esimerkiksi viiteen etappiin. Jokainen etappi testaa eri osa-alueita:

Vaihe 1: Kaupunkiajo (Hidas vauhti, vähän ilmavirtaa)

- **Testi:** Jäähdytysjärjestelmä + Kaasutin (tyhjäkäynti).
- **Logiikka:** Jos `cooling_quality < 60`, lämpötila nousee +20 astetta. Jos se ylittää 100, pyörä alkaa keittää.
- **Kerronta:** *"Valot vaihtuvat punaisiksi. Huomaat, kuinka moottorista nouseva lämpö alkaa polttaa reisiäsi. Onko tuuletin päällä?"*

Vaihe 2: Moottoritie (Kovat kierrokset, pitkäkestoinen rasitus)

- **Testi:** Moottorin alakerta + Voimansiirto.
- **Logiikka:** Peli heittää satunnaisluvun suhteessa laatuun. `if renpy.random.randint(0, 100) > engine_quality: "Kiertokanki alkaa nakuttaa."`
- **Kerronta:** *"Moottori ulvoo satasen vauhdissa. Tunnet jalkatapeissa oudon, rytmisen tärinän, joka ei kuulu asiaan..."*

---

3. "Tienvarsikorjaus" – Pelaajan pelastusrengas

Jos jokin osa-alue alkaa pettää (esim. lämpö nousee liikaa tai pultti löystyy), peli ei heti lopu. Pelaajalle annetaan mahdollisuus **tienvarsikorjaukseen**:

- **Pysähdy jäähdyttämään:** Laskee lämpöä, mutta kuluttaa aikaa (saatat myöhästyä periltä).
- **Kiristä pultteja hätäisesti:** Mahdollisuus korjata värinää, mutta vaatii "työkalu-checkin" (onko muistettu ottaa mukaan jakoavain?).
- **Paina menemään:** Riski kasvaa, mutta ehkä ehdit perille ennen kuin moottori räjähtää.

---

4. Matkalogiikan koodiesimerkki (Python)

Tämä funktio voisi laskea matkan onnistumisen:

Python

```
init python:
    def laske_matkan_tulos(osat):
        # osat on lista Työkohde-objekteja
        vauriot = 0
        yla_lampo = 0

        for o in osat:
            if o.laatu < 50: # Jos laatu on huono
                vauriot += 1
            if o.nimi == "Jäähdytysjärjestelmä":
                yla_lampo = 100 - o.laatu # Mitä huonompi laatu, sitä enemmän lämpöä

        if vauriot > 3 or yla_lampo > 50:
            return "breakdown" # Pyörä hajoaa matkalle
        elif vauriot > 0:
            return "limp_home" # Pääset perille, mutta pyörä on romuna
        else:
            return "perfect" # Täydellinen suoritus
```

---

5. Visuaalinen toteutus Ren'Py-moottorilla

Monotonisuuden tappamiseksi matkan aikana käytetään **näkökulman vaihtelua**:

1. **Dashboard-näkymä:** Näytä pelaajalle mittarit (lämpö, nopeus).
2. **Lähikuvat:** Jos ketjut löystyvät, näytä kuva tärisevistä ketjuista.
3. **Ääni:** Tämä on kriittinen! Lisää taustalle moottorin ääni, johon sekoittuu "kilinää" tai "suhinaa", jos jokin on pielessä.

Lopputulokset

- **Täydellinen ajo:** Pääset perille, pyörä on loistokunnossa. Saat "Legendan maineen".
- **Selviytyminen:** Pääset perille savuttaen ja paukuttaen. Pyörä pitää rakentaa uusiksi, mutta tavoite saavutettu.

- **Keskeytys:** Jäät tien varteen. Peli näyttää, mikä osa petti (esim. *"Syy: Murtunut öljyletku, laatu 32%"*). Tämä opettaa pelaajaa seuraavaa yritystä varten.

Haluaisitko, että hahmottelen tuon dashboardin (mittariston) screen-koodin, joka päivittyy matkan edetessä?

Juu kerro dashboardin screen-koodista enemmän

Dashboard eli mittaristo on matka-osion sydän. Sen tehtävä on viestiä pelaajalle pyörän tila ilman, että sitä tarvitsee lukea pelkästä tekstistä. Ren'Py:ssä tämä toteutetaan screen-komennolla ja bar-elementeillä.

Tässä on syväsukellus dynaamiseen mittaristoon.

---

1. Mittariston visuaalinen logiikka

Mittaristo koostuu kolmesta pääasiasta:

- **Lämpömittari (Heat Gauge):** Nousee, jos seokset ovat laihat tai jäähdytys ontuu.
- **Kestävyyspalkki (Integrity/Condition):** Laskee, jos pyörä tärisee liikaa tai osat pettävät.
- **Matkamittari (Distance):** Näyttää, kuinka lähellä maali on.

---

2. Screen-koodi (Ren'Py)

Tämä koodi luo mittariston, joka päivittyy automaattisesti muuttujien muuttuessa.

Koodinpätkä

```
screen dashboard_view(vauhti, lampo, kunto, matka):
    # Luodaan kiinteä kehys mittaristolle ruudun alaosaan
    frame:
        xalign 0.5
        yalign 0.95
        background Frame("gui/dashboard_bg.png", 10, 10) # Taustakuva mittaristolle
        padding (20, 20)

        hbox:
            spacing 50

            # LÄMPÖMITTARI
            vbox:
                text "MOOTTORIN LÄMPÖ" size 16 xalign 0.5
                bar value lampo range 120: # 120 astetta on kriittinen raja
                    xsize 200
                    # Muuttaa väriä: jos lampo > 100, palkki on punainen
```

```
            left_bar Frame(ConditionSwitch(
                "lampo < 90", "#00ff00",
                "lampo < 105", "#ffaa00",
                "True", "#ff0000"
            ), 10, 10)

        # MATKAMITTARI
        vbox:
            text "MATKA PERILLE" size 16 xalign 0.5
            bar value matka range 100:
                xsize 300
            text "[matka] / 100 km" size 14 xalign 0.5

        # KUNTO/VÄRINÄ
        vbox:
            text "PYÖRÄN KUNTO" size 16 xalign 0.5
            bar value kunto range 100:
                xsize 200
                left_bar "#0080ff" # Sininen vakauden merkki
```

---

3. Matka-simulaation logiikka (Python)

Tämä on se dynaaminen osuus, joka pyörii taustalla matkan aikana. Se laskee joka sekunti, miten pyörä reagoi.

Python

```
init python:
    def matka_simulaatio():
        # Globaalit muuttujat, joita dashboard lukee
        global matka_edistys, moottorin_lampo, pyoran_kunto

        # PERUSLOGIIKKA:
        # Lämpö nousee, jos kaasutin on säädetty huonosti (kaasutin_laatu)
        # Lämpö = Peruslämpö + (Nopeus / Jäähdytyslaatu)

        perus_lampo = 80
        lampo_nousu = (nykyinen_vauhti * 0.5) / (jaahdytys_laatu / 100.0)
        moottorin_lampo = perus_lampo + lampo_nousu

        # Jos tärinä on kovaa, kunto laskee hiljalleen
        if alakerta_laatu < 60:
            pyoran_kunto -= 0.1 * (nykyinen_vauhti / 50)
```

---

4. Miten vältetään monotonisuus matkan aikana?

Pelkkä mittarin tuijotus on tylsää. Tuodaan siihen **interaktiivisuutta** ja **audiovisuaalista palautetta**:

1. **Screen Shaking:** Jos `pyoran_kunto` laskee tai `alakerta_laatu` on huono, käytetään Ren'Py:n `hpunch` tai `vpunch` komentoja satunnaisesti:
2. Koodinpätkä

```
if pyoran_kunto < 50 and renpy.random.randint(1, 10) == 1:
    with hpunch
    "Pyörä tärisee pelottavasti!"
```

3.
4.
5. **Äänen muutos:** Käytetään `renpy.music.set_volume` muuttamaan moottorin äänen sävyä. Jos lämpö nousee yli 105, lisätään taustalle korkea suhina (kiehuva jäähdytysneste).
6. **Tienvarsivalinnat:** Mittaristoon voi ilmestyä nappi "HÄTÄPYSÄYTYS", kun lämpö nousee liikaa.
7. Koodinpätkä

```
if moottorin_lampo > 110:
    textbutton "PYSÄHDY JÄÄHDYTTÄMÄÄN":
        action Jump("tienvarsi_jaahdytys")
        background "#ff0000"
```

8.
9.

Matkan loppuhuipentuma

Kun `matka_edistys` saavuttaa 100, tarkistetaan `pyoran_kunto`.

- Jos se on > 80: "Pyörä kesti kuin unelma!"
- Jos se on 20-80: "Pääsit perille, mutta moottori taitaa vaatia täysremontin."
- Jos se on < 20: "Pyörä suli kasaan viimeisellä kilometrillä."

Tämä dashboard tekee korjausvaiheen valinnoista merkityksellisiä. Haluaisitko, että rakennamme seuraavaksi sen mekaniikan, jolla pelaaja voi matkan aikana yrittää "paikata" huonoja säätöjään (esim. rikastamalla seosta lennosta)?

Joo tienvarsi remontti kiinnostaa

Tienvarsiremontti on pelimekaniikkana loistava, koska se muuttaa aiemmat "epäonnistumiset" tallissa uusiksi pelillisiksi haasteiksi. Se antaa pelaajalle mahdollisuuden pelastaa tilanteen, vaikka jokin osa-alue olisi tehty huonosti.

Tässä on ehdotus siitä, miten **tienvarsiremontti** toimii teknisesti ja kerronnallisesti:

---

1. Remontin käynnistyminen

Pelaaja voi joutua tien varteen kahdella tavalla:

1. **Vapaaehtoinen pysähdys:** Pelaaja huomaa mittaristosta (dashboard), että lämmöt nousevat, ja päättää pysähtyä ennen kuin jotain hajoaa.
2. **Pakollinen hätäpysäytys:** Peli heittää "Breaking Point" -tapahtuman, jos jonkin osan laatu on alle kriittisen rajan. *"Moottori alkaa paukuttaa – sinun on pakko vetää sivuun heti!"*

2. Resurssit: Työkalut ja tarvikkeet

Tässä kohtaa aamulla tehty valmistelu nousee arvoonsa. Pelaajan kannalta on kriittistä, mitä hän otti mukaan:

- **Työkalusarja:** Mahdollistaa oikeat korjaukset. Ilman tätä voit vain "potkia ja toivoa".
- **Nippusiteet ja ilmastointiteippi:** "Purkkaviritysten" kuningasvarusteet.
- **Varaöljy tai vesi:** Auttaa ylikuumenemiseen tai vuotoihin.

---

3. Logiikka: "Purkkaviritys" vs. "Oikea korjaus"

Kun pelaaja on tien varressa, avautuu uusi valikko. Tässä on esimerkkikoodia siitä, miten valinnat vaikuttavat pyörän tilaan matkan jatkuessa:

Python

```
init python:
    def tienvarsi_korjaus(kohde, tapa, tyokalut=False):
        # kohde = viallinen osa (esim. jäähdytin)
        # tapa = "hata" tai "huolellinen"

        if tapa == "hata": # Nippuside-viritys
            kohde.laatu += 20
            matka_aika += 10 # Vie vähän aikaa
            # Purkkaviritys on epävakaa:
            return "Pysyy kasassa... ehkä. Jatka matkaa."

        elif tapa == "huolellinen" and tyokalut:
            kohde.laatu += 50
            matka_aika += 45 # Vie paljon aikaa, aurinko laskee
            return "Kunnollinen korjaus tehty. Voit luottaa tähän paremmin."
```

---

4. Jännitysmomentti: Aika ja ympäristö

Tienvarsiremontti ei tapahdu tyhjiössä. Voit lisätä painetta seuraavilla tavoilla:

- **Auringonlasku:** Jos matka viivästyy liikaa, alkaa hämärtää. Remontti vaikeutuu (vaikeuskerroin nousee), ja tarvitset taskulampun nähdäksesi laskurin.

- **Ohi kulkevat ihmiset:** Kuten tallissakin, joku voi pysähtyä "auttamaan" (eli häiritsemään ja viemään aikaa).
- **Lika:** Kädet menevät rasvaan. Jos et puhdista niitä, seuraava työvaihe on liukkaampi ja vaikeampi.

---

5. Visuaalinen toteutus Ren'Py-ruudulla

Voit käyttää tallista tuttua laskuri-mekaniikkaa, mutta muuta visuaalit "kenttäolosuhteisiin":

Koodinpätkä

```
screen roadside_fix_ui(kohde):
    add "images/bg_roadside_ditch.png" # Taustalla ojanpiennar

    vbox:
        align (0.5, 0.2)
        text "KORJATAAN: [kohde.nimi]" size 40
        text "Kunto nyt: [kohde.laatu]%" color "#ff0000"

    hbox:
        align (0.5, 0.8)
        spacing 20
        textbutton "Nippuside-viritys":
            action Return("hata")
            hover_background "#ffaa00"

        if has_tools:
            textbutton "Pura ja kiristä (vaatii työkaluja)":
                action Return("huolellinen")
                hover_background "#00ff00"
        else:
            textbutton "Pura ja kiristä":
                action None # Harmaa nappi, jos ei työkaluja
                text_color "#555"
```

---

6. "Purkkavirityksen" kerronnallinen seuraus

Pelin loppumatkalla voit antaa pelaajalle palautetta näistä hätäpaikkauksista.

- Jos pyörä kestää perille asti teipin voimalla: *"Saavut perille moottori rouskuttaen. Ensimmäinen nippuside suli juuri, kun kurvasit pihaan. Se oli sekunneista kiinni!"*
- Tämä luo pelaajalle mahtavan onnistumisen tunteen ("Selvisin vastoinkäymisistä!").

Yhteenveto koko loopista:

1. **Aamu:** Valitse mitä korjaat ja kuinka hyvin (10 konttia).
2. **Valmistelu:** Pakkaa työkalut (riski: paino vs. hyöty).

3. **Matka:** Seuraa dashboardia.
4. **Kriisi:** Pysähdy tien varteen korjaamaan, jos jokin pettää.
5. **Finaali:** Selviätkö perille vai jäätkö pimeään metsään?

Miltä tämä kokonaisuus vaikuttaa? Onko mekaniikassa jokin kohta, joka tuntuu vielä liian vaikealta tai jota pitäisi selventää (esim. miten työkalujen pakkaaminen vaikuttaa matkaan)?

Joo kerro työkalujen pakkaamisesta. Idea on että työkalut, pyörän varaosat ja matkatavarat (teltta, ruokailuvälineet, kaljakori) eivät kaikki mahdu kerralla kyytiin. Reissun lähdettäessä pitää tehdä valintaa. Esim varakaasutin, lataus säädit, puola tai sytytysmoduli olis hyvä olla mukana mutta pelaajan pitää tehdä sorttaus ennen matkaa.

Tämä "pakkaus-dilemma" on erinomainen lisä pelimekaniikkaan, koska se pakottaa pelaajan arvioimaan riskejä. Se tekee tallissa tehdyistä valinnoista entistä painavampia: *”Luotanko niin paljon tallissa tekemääni kaasuttimen säätöön, että jätän varakaasuttimen kotiin ja otan tilalle kaljakorin?”*

Tässä on logiikka ja toteutustapa Ren’Py-moottorilla:

1. Reppubudjetti: Tila ja paino

Määritellään pyörän kantokyky (esim. ”Tilavuusyksiköt”). Jokaisella esineellä on oma kokonsa.

Python

```
init python:
    class Esine:
        def __init__(self, nimi, koko, tyyppi, kuvaus):
            self.nimi = nimi
            self.koko = koko  # Kuinka paljon tilaa vie (1-50)
            self.tyyppi = tyyppi # "tyokalu", "varaosa" tai "hupi"
            self.kuvaus = kuvaus

    # Kantokyky
    pyoran_tila = 100
    pakatut_esineet = []

    # Esimerkki-esineitä
    tavarat = [
        Esine("Hylsysarja", 30, "tyokalu", "Kattava setti, korjaa melkein mitä vain."),
        Esine("Varakaasutin", 20, "varaosa", "Jos vanha alkaa tulvia pahasti."),
        Esine("Sytytysmoduuli", 10, "varaosa", "Pieni mutta kriittinen. Sähkövika jättää tielle."),
        Esine("Kaljakori", 50, "hupi", "Matkan kohokohta, mutta vie puolet tilasta."),
        Esine("Teltta ja pussi", 40, "hupi", "Mahdollistaa nukkumisen (palauttaa energiaa)."),
        Esine("Nippuside-pussi", 5, "tyokalu", "Kevyt hätäapu kaikkeen.")
    ]
```

2. Pakkausruutu (The Sorting Screen)

Pelaajalle näytetään kaksi listaa: ”Tallin hylly” ja ”Pyörän kyyti”.

Logiikka:

- Pelaaja klikkaa esinettä hyllyltä -> se siirtyy pyörään, jos tila riittää.
- Pelaaja klikkaa esinettä pyörästä -> se palaa hyllylle.
- Ruudulla näkyy palkki, joka täyttyy sitä mukaa kun tavaraa lisätään.

3. Valintojen seuraukset (The Trade-offs)

Tämä on pelin dynaamisin osa. Pakkausvalinnat vaikuttavat suoraan matkan kulkuun:

Valinta	Hyöty	Haitta / Riski
Varaosat (esim. puola)	Jos osa petti matkalla, voit vaihtaa sen uuteen tien varressa ja matka jatkuu 100% laadulla.	Vie tilaa teltalta tai kaljalta.
Pelkät työkalut	Voit yrittää korjata vanhan osan (laatu nousee esim. 30% -> 50%).	Vanha osa voi pettää uudestaan pian.
Hupi/Mukavuus (teltta/olut)	Parempi loppupisteytys, hahmon stressi laskee, "Good Ending" mahdollisuus.	Jos pyörä hajoaa, et voi tehdä mitään. Joudut soittamaan hinauksen (Game Over).

4. Esimerkkitilanne matkan aikana

Koodi tarkistaa, onko tarvittava esine mukana, kun ongelma ilmenee:

Koodinpätkä

```
label moottori_pätkii:
    "Moottori alkaa pätkiä ja sammuu lopulta kokonaan. Tutkit tilannetta..."

    if "Sytytysmoduuli" in pakatut_esineet:
        menu:
            "Sinulla on varamoduuli mukana! Vaihda se?":
```

```
            $ pyora.cdi_laatu = 100
            "Vaihdat moduulin kymmenessä minuutissa. Matka jatkuu!"
    else:
        "Toteat moduulin kärähtäneen. Sinulla ei ole varaosaa mukana."
        if "Hylsysarja" in pakatut_esineet:
            "Yrität puhdistaa liittimiä työkaluilla, mutta se on vain tekohengitystä..."
            $ pyora.cdi_laatu = 30
        else:
            "Sinulla ei ole edes työkaluja. Matka tyssäsi tähän."
            jump game_over_hinaus
```

5. "Kaljakori-muuttuja" ja loppuratkaisu

Voit lisätä peliin huumoria ja tavoitteellisuutta näillä hupitavaroilla:

- **Kaljakori:** Jos se on mukana perillä, saat "Bileet!"-lopetuksen.
- **Teltta:** Jos joudut yöpymään tien varressa korjaamassa, teltta estää hahmoa tulemasta kipeäksi/väsyneeksi.
- **Ilman telttaa:** Hahmo nukkui ojassa, seuraavana päivänä laskuri liikkuu 50% hitaammin ja virheitä tulee enemmän.

Miten tästä saa visuaalisesti hienon?

Ren'Py:ssä voit käyttää drag and drop -toimintoa, jossa pelaaja fyysisesti vetää kaljakorin pyörän tarakalle. Jos kori on liian iso, se "pomppaa" takaisin hyllylle ja kuuluu virheääni.

Haluatko, että katsotaan, miten tuo "tilan loppuminen" ilmoitetaan pelaajalle selkeästi, tai miten esineiden kuvakkeet kannattaisi sijoitella?

Kun tehdään yhtä osa-aluetta tulee vastaan 10 kysymystä miten tämän työvaiheen suoritat. Yksi on se täysin oikea ja yleensä hitain. Tuurilla ja tarkkana ollessa jokin homma onnisuu iha hyvin vaikka vetää riskillä. Väsyneenä ja stressissä se oikein tekeminenkään ei ole helppoa.

Tämä on erinomainen tapa syventää pelattavuutta! Se muuttaa "työskentely-loopin" passiivisesta odottelusta aktiiviseksi päätöksenteoksi. Koska jokaisessa 10 työkontissa on 10 työvaihetta (yhteensä 100 pientä päätöstä!), meidän on rakennettava järjestelmä, joka on hallittava sekä sinulle kehittäjänä että pelaajalle.

Tässä on logiikka, miten nämä **10 kriittistä valintaa** per työvaihe toteutetaan:

---

1. Työvaiheen rakenne: "Hitaasti hyvä tulee vs. Riski"

Jokainen valinta vaikuttaa kahteen asiaan: **Laatuun** ja **Aikaan**.

Valintatyyppi	Ajan kuluminen	Laadun muutos	Riski (Epäonnistuminen)
Oppikirjan mukaan	Suuri (-20 min)	+10 (Varma)	Erittäin matala
Sinnepäin (Rutiini)	Normaali (-10 min)	+5	Pieni (Väsymys vaikuttaa)
Oikotie (Riski)	Pieni (-2 min)	+15 tai -30	Korkea (Vaatii tuuria/tarkkuutta)

---

2. Väsymys ja Stressi "vaikeuskerroinvaihtoehtoina"

Hahmon tila ei vain kerro tekstiä, vaan se muuttaa valintojen onnistumistodennäköisyyksiä.

- **Stressi:** Pienentää "Oikean valinnan" tuottamaa laatuvoittoa (kädet tärisevät, pultti ei mene kierteille).
- **Väsymys:** Kasvattaa "Oikotien" riskiä eksponentiaalisesti.

---

3. Tekninen toteutus (Ren'Py Python-logiikka)

Jotta sinun ei tarvitse kirjoittaa 100 erillistä valikkoa, tehdään **"Valintageneraattori"**, joka hakee tiedot listasta.

Python

```
init python:
    # Esimerkki yhden kontin (esim. Kaasutin) 10 vaiheesta
    kaasutin_vaiheet = [
        {"q": "Puhdistatko suuttimet?", "opt1": "Ultraäänipesu (Hidas)", "opt2": "Puhallus ja rukoilu (Riski)"},
        {"q": "Kohon säätö:", "opt1": "Mittaa tönärillä (Tarkka)", "opt2": "Taivuta sormin (Nopea)"},
        # ... loput 8 vaihetta
    ]

    def laske_tulos(valinta, pelaaja):
        onni = renpy.random.randint(1, 100)
        # Väsymys-sakko: jos energia on 20, sakko on 40 pistettä
        sakko = (100 - pelaaja.energia) * 0.4

        if valinta == "tarkka":
```

```
        # Oikea tapa onnistuu lähes aina, mutta väsymys voi pilata senkin
        if onni > (5 + sakko): return "Hyvä"
        else: return "Hutiloitu"

    if valinta == "riski":
        # Riski vaatii kovaa onnea
        if onni > (60 + sakko): return "Jättipotti"
        else: return "Pieleen"
```

---

4. Pelin kulku (Screen-näkymä)

Kun pelaaja valitsee "Kaasutin-kontin", peli käy läpi loopin:

Koodinpätkä

```
label tyovaihe_loop(kontti_data):
    $ vaihe_nro = 0
    $ kokonais_laatu = 0

    while vaihe_nro < 10:
        $ v = kontti_data[vaihe_nro]
        "Vaihe [vaihe_nro + 1] / 10: [v[q]]"

        menu:
            "[v[opt1]]": # Hidas ja varma
                $ tulos = laske_tulos("tarkka", pelaaja)
                $ kuluva_aika += 20
            "[v[opt2]]": # Nopea ja riski
                $ tulos = laske_tulos("riski", pelaaja)
                $ kuluva_aika += 5

        # Päivitetään laatu tuloksen mukaan
        if tulos == "Hyvä": $ kokonais_laatu += 10
        elif tulos == "Jättipotti": $ kokonais_laatu += 15
        elif tulos == "Pieleen": $ kokonais_laatu -= 20

        $ vaihe_nro += 1

    return kokonais_laatu
```

---

5. Miten vältetään monotonisuus 10 vaiheen aikana?

10 kysymystä peräkkäin voi alkaa tuntua kuulustelulta. Tässä 3 kikkaa sen rikkomiseen:

1. **Visuaalinen edistyminen:** Joka 2. kysymyksen jälkeen näytetään välianimaatio tai kuva, jossa moottoripyörä muuttuu (esim. osia kiinnittyy).

2. **Hahmon sisäinen monologi:** Jos stressi on korkea, hahmo alkaa kesken kysymysten mutisemaan: *"Hitto, miksi otin tämän urakan... särkee päätä."* Tämä antaa pelaajalle vihjeen, että ehkä kannattaa valita se hitain/varmin vaihtoehto.
3. **Kriittinen virhe -keskeytys:** Jos jokin vaihe menee täysin "Pieleen", työvaihe voi keskeytyä: *"Pultti katkesi sisään. Sinun on pakko käyttää loppupäivä sen poraamiseen ulos."* (Tämä opettaa pelaajalle, että riskeillä on oikea hinta).

Seuraava askel

Kun nämä 10 vaihetta on käyty läpi, saadaan se **lopullinen laatuarvo** (esim. 78/100). Tämä arvo siirtyy suoraan siihen aiemmin puhumaamme **dashboardiin** ja vaikuttaa matkan onnistumiseen.

Haluaisitko, että mietimme tarkemmin niitä "Pieleen"-tulosten kerronallisia seurauksia? Esimerkiksi, miten peli kuvailee sitä, kun pelaaja väsyneenä yrittää vetää riskillä ja jokin menee rikki?

Joo kerro lisää pieleen menevien tulosten kerronnallisia vaikutuksia

Kun työvaihe menee "päin mäntyä", se ei ole pelkkä numeroarvon lasku, vaan se on pelillinen ja kerronnallinen käännekohta. Se luo draamaa: pelaaja tietää tehneensä virheen, ja se jää kummittelemaan takaraivoon koko matkan ajaksi.

Tässä on eri tasoisia kerronnallisia vaikutuksia "Pieleen"-tuloksille:

---

1. Välittömät seuraukset tallissa (Immediate Drama)

Kun laskuri pysähtyy ja tulos on katastrofi, peli voi käyttää visuaalisia ja auditiivisia keinoja korostamaan epäonnistumista:

- **Ääniefekti:** Metallinen *KLANK*, pultin katkeamisen napsahdus tai nesteen loiske lattialle.
- **Visuaalinen palaute:** Ruutu heilahtaa (shake), hahmon kädet peittyvät mustaan öljyyn tai ruutu välähtää punaiseksi.
- **Hahmon reaktio:** *"Ei jumalauta... se kierre meni sileäksi. Se oli siinä."* tai *"Kädet tärisee niin paljon, että hylsy liukui ja rystyset aukesi."*
- **Mekaaninen lukitus:** Kyseinen työvaihe saattaa lukittua. Et voi enää yrittää "Täydellistä", vaan joudut tekemään "Paikkauksen", joka vie loppupäivän tunnit.

2. Piilevät viat (The "Ominaisuus" effect)

Joskus pelaaja ei heti tiedä, kuinka pahasti jokin meni pieleen. Tämä luo jännitystä matkalle:

- **Epävarmuus:** Peli ilmoittaa: *"Sait sen kasaan, mutta pultti jäi vähän löysälle. Ehkä se kestää... ehkä ei."*

- **Kerronnalinen muistutus:** Myöhemmin illalla hahmo pohtii: *"Se kaasuttimen koho jäi vähän vinoon. Toivottavasti se ei tulvi hämärässä."*
- **Visualisointi tallissa:** Jos teet työn huonosti, pyörän ulkonäkö voi muuttua. Esim. tiivisteen välistä tihkuu öljyä jo tallin lattialle, tai jokin osa repsottaa hieman vinossa.

3. Ketjureaktiot (Stress and Spiral)

Virheillä on tapana kasvaa korkoa. Jos yksi vaihe menee pieleen, hahmon tila muuttuu:

- **Stressi-piikki:** Epäonnistuminen nostaa stressiä +20. Tämä tekee seuraavista 9 työvaiheesta vaikeampia, koska "oikein tekemisen" ikkuna pienenee.
- **Kiroilumittari:** Jos pelaaja epäonnistuu usein, hahmon monologi muuttuu aggressiivisemmaksi. Valinnat valikoissa voivat muuttua: "Yritä varovasti" muuttuu muotoon "Survo se paikalleen väkisin".

4. Matkan aikaiset kerronnaliset "flashbackit"

Kun pyörä lopulta pettää tien päällä, peli palauttaa pelaajan mieleen sen nimenomaisen hetken tallissa:

- **Kohtalon hetki:** *"Muistat sen sateisen tiistai-aamun, kun väsyneenä kiristit tätä nimenomaista pulttia. Nyt se makaa jossain asfalttiauran hampaissa, ja takarengas on jumissa."*
- **Huumori ja ironia:** Jos otit kaljakorin mukaan varaosien sijasta: *"Katsot tyhjää työkalulaukkua ja sitten kaljakoria tarakalla. Olut on kylmää, mutta moottori on kylmempi. Ainakaan ei tarvitse kotiin kävellessä janoon kuolla."*

Esimerkki: Katkennut pultti -skenaario (Koodi/Logiikka)

Tämä on klassinen mekaanikon painajainen, joka voidaan toteuttaa Ren'Py-menuna:

Koodinpätkä

```
label pultti_katkeaa:
    play sound "audio/bolt_snap.ogg"
    with hpunch
    "KUULUU TERÄVÄ NAPS."
    pelaaja "Voi helvetin helvetti..."
    "Yritit säästää aikaa ja kiristit pultin ilman momenttiavainta. Se katkesi sisään."

    $ stressi += 30
    $ jaahdytys_laatu -= 40
    $ tyotunnit_jaljella -= 2 # Pultin ulosporaaminen vie aikaa

    menu:
        "Poraa pultti ulos (vie 2 tuntia, nostaa stressiä)":
            "Hikoilet ja kiroilet. Lopulta saat jämät pois, mutta kierteet ovat entiset."
            $ kierteet_ehjat = False
        "Jätä se silleen ja käytä liimaa (Riski matkalla)":
```

"Pyyhit öljyt pois ja vedät päälle kerroksen kemiallista metallia. Päältä päin sitä ei huomaa."
$ purkkaviritys_aktiivinen = True

5. Pieleen-tuloksen vaikutus Dashboardiin

Dashboardilla (mittaristossa) huono laatu näkyy epävakautena:

- **Heiluvat viisarit:** Lämpömittari ei pysy paikallaan, vaan "nypyttää".
- **Varoitusvalot:** Jos teit sähköt huonosti, Dashboardilla vilkkuu satunnaisesti latausvalo, vaikkei mitään hätää olisikaan (tai onko?).

Miltä tämä kerronnallinen syvyys kuulostaa? Haluaisitko, että kehitämme noita "purkkavirityksiä" lisää – eli miten pelaaja voi yrittää huijata peliä (ja itseään) korjausvaiheessa?

Sitten matka-ajosta. Suoraan lyhyttä tietä, kova nopeus ja kovan nopeuden ongelmat. Kierrellen hitaasti sivuteitä turvallisesti ja hitaasti. Oikoen kärrypolkua pitkin milloin kaikenlaiset riskit mahdollisia ja suuri ajansäästö.

Tämä reittivalinta-mekaniikka on loistava tapa sitoa tallissa tehdyt päätökset lopulliseen koitokseen. Se muuttaa matkan pelkäksi "selviytymistaisteluksi", jossa pelaaja joutuu arvioimaan pyöränsä kuntoa suhteessa valittuun riskiin.

Tässä on ehdotus siitä, miten nämä kolme eri etenemistapaa vaikuttavat pelilogiikkaan ja kerrontaan:

---

1. Reittivalinta-valikko (The Navigator)

Matka voidaan jakaa useampaan etappiin (esim. 3–5 osaa), ja jokaisen alussa pelaaja valitsee strategiansa.

Esimerkki valinnasta:

- **Valtatie (Fast & Furious):** *"Vedän urku auki suoraan perille."*
- **Sivutiet (The Tourist):** *"Nautiskelen maisemista ja otan iisisti."*
- **Metsäpolku (The Shortcut):** *"Oikaisen tästä rääseikön läpi, säästän tunteja."*

---

2. Reittien vaikutus pyörän muuttujiin

A. Valtatie: Suora tie, kova nopeus

Tämä reitti testaa moottorin sisäistä kestävyyttä ja lämmönhallintaa.

- **Kriittiset osat:** Moottorin alakerta, Jäähdytys, Kaasuttimen säädöt (laiha seos kuumentaa moottoria).

- **Riski:** Jos teit alakerran tai sytytyksen huonosti, kova kierrosluku räjäyttää moottorin nopeasti.
- **Kerronta:** *"Moottori ulvoo ja tuuli repii kypärää. Tie on tylsä, mutta vauhti on kova. Huomaat lämpömittarin neulan hiipivän kohti punaista..."*

B. Sivutiet: Kierrellen ja hitaasti

Turvallisin valinta, jos tiedät, että pyörässä on "purkkavirityksiä".

- **Kriittiset osat:** Mukavuus, Jarrut (mutkainen tie), Yleinen luotettavuus.
- **Riski:** Aika loppuu kesken. Jos aurinko laskee tai kalja lämpenee, häviät pelin kerronnallisesti.
- **Kerronta:** *"Kauniita peltoja ja leppoisaa mutkittelua. Pyörä tuntuu pysyvän kasassa, mutta kello käy armottomasti. Joudutko ajamaan loppumatkan pimeässä?"*

C. Kärrypolku: Riskiä ja oikomista

Testaa pyörän fyysistä kestävyyttä ja pulttien kireyttä.

- **Kriittiset osat:** Jousitus, Runko, Renkaat ja ennen kaikkea se, kiristitkö kaikki pultit.
- **Riski:** Tärinä irrottaa osia. Jos jousituspultti on "hutiloitu", se katkeaa täällä.
- **Kerronta:** *"Kivet sinkoilevat ja pyörä hyppii nimismiehenkiharoissa. Tunnet jokaisen täräyksen hampaissasi. Yhtäkkiä kuuluu metallinen kilahdus – tippuko sieltä jotain?"*

---

3. Matkalogiikka Ren'Py-koodina

Voimme käyttää "vaikeuskerrointa" (modifier), joka kertoo tallissa tehdyn laadun reitin haasteilla.

Python

```
init python:
    def suorita_etappi(reitti, pyora):
        vaurio_maara = 0

        if reitti == "valtatie":
            # Valtatie rankaisee lämmöstä ja alakerrasta
            if pyora.jaahdytys_laatu < 70:
                vaurio_maara += (70 - pyora.jaahdytys_laatu) * 2
            aika_kuluu = 30 # Nopea

        elif reitti == "sivutie":
            # Sivutie on armollinen, mutta hidas
            vaurio_maara += 5
            aika_kuluu = 120 # Todella hidas

        elif reitti == "karrypolku":
            # Kärrypolku testaa pultit ja jousituksen
            if pyora.runko_laatu < 80:
                vaurio_maara += (80 - pyora.runko_laatu) * 3
```

```
        aika_kuluu = 45 # Oikopolku

    return vaurio_maara, aika_kuluu
```

---

4. Kerronnalliset huipentumat reitin varrella

Jotta matka tuntuu dynaamiselta, peli tekee "tarkistuspisteitä" (checkpoints):

- **Valtatiellä:** *"Olet ohittamassa rekkaa, kun moottori alkaa pätkiä. Uskallatko pitää kaasun pohjassa vai päästätkö irti? (Vaatii Sytytysmoduulin laadun tarkistuksen)"*
- **Kärrypolulla:** *"Syvä lätäkkö edessä. Jos jousitus on säädetty liian pehmeäksi, pohjaat ja öljypohja voi haljeta. Kierrätkö (vie aikaa) vai ajatko läpi (riski)?"*

5. Aikapaine ja "Kaljakori-indeksi"

Pelaajalla on kiinteä aika päästä perille (esim. 8 tuntia).

- Jos valitset vain **sivuteitä**, peli loppuu siihen, että saavut pimeään metsään ilman valoja.
- Jos otat **kärrypolun**, säästät aikaa, jolloin voit pitää perillä pidemmät juhlat (parempi loppupisteytys).
- Jos **valtatiellä** pyörä alkaa ylikuumentua, joudut ehkä valitsemaan hitaamman reitin loppumatkaksi, mikä sekoittaa aikataulusi.

Miltä tämä kuulostaa?

Tämä luo matkasta taktisen palapelin. Pelaajan täytyy miettiä: *"Tein sähköt huonosti, mutta pultit on kireällä. En uskalla mennä valtatielle, mutta kärrypolku saattaisi toimia, koska tärinä ei vaikuta sähköihin niin pahasti kuin jatkuva kuumuus."*

Haluaisitko, että hahmottelen sen "kohtalon hetken", kun pelaaja tajuaa valinneensa väärän reitin pyörän kuntoon nähden? (Esim. sähkövika vesisateessa sivutiellä tai kiinnileikkaaminen motarilla.)

Joo. ja idea on että matkalla tehtään useita valintoja. Reitti on kyllä jo etukäteen selvä mutta oikomalla on mahdollista päästä luotettavalla pyörällä nopeammin seuraavaan valintapisteeseen.

Tämä muuttaa pelin rakenteen mielenkiintoiseksi "riskinhallintamatkaksi". Reitti on kiinteä, mutta etenemisnopeus ja riskitaso vaihtelevat pelaajan lennossa tekemien valintojen mukaan.

Kuvitellaan matka sarjana **etapeja**. Jokaisen etapin lopussa on valintapiste, jossa päätetään, miten seuraava osuus suoritetaan.

---

1. Etappirakenne ja valintapisteet

Matka koostuu esimerkiksi viidestä segmentistä (esim. Kotikylä -> Huoltoasema -> Jokiylitys -> Vuorisola -> Päätepiste).

Valinta	Kuvaus	Vaikutus pyörään	Aikahyöty
Normaali reitti	Tasainen eteneminen.	Peruskulutus ja pieni lämmönnousu.	+/- 0 min
Kova kiire (Hana auki)	Puristetaan moottorista kaikki irti.	**Lämpö nousee**, alakerta kovilla.	-15 min
Oikopolu (Kärrytie)	Lyhyempi matka, mutta huono alusta.	**Tärinä kasvaa**, pultit ja jousitus kovilla.	-30 min

---

2. Oikopolun mekaniikka: "Luotettavuus-check"

Oikopolku on houkutteleva, koska se säästää eniten aikaa, mutta se on myös vaarallisin. Se testaa suoraan tallissa tehtyä työtä.

Esimerkkilogiikka oikopolulle:

1. **Vaatimus:** Pyörän `runko_laatu` ja `pulttien_kireys` on oltava > 70.
2. **Tapahtuma:** Jos laatu on alle rajan, heitetään "noppaa". Jos nopat epäonnistuvat, peli hyppää **tienvarsiremontti-labeliin**.
3. **Palkinto:** Pelaaja saavuttaa seuraavan pisteen huomattavasti nopeammin, mikä antaa pelivaraa myöhempiin ongelmiin.

---

3. Matkakartan Screen-toteutus

Pelaajalle kannattaa näyttää visuaalinen kartta jokaisessa valintapisteessä, jotta hän näkee etenemisensä ja kuluvan ajan.

Koodinpätkä

```
screen matka_kartta(etappi, aikaa_jaljella):
    frame:
        xalign 0.5 yalign 0.1
        hbox:
```

```
        spacing 20
        text "Etappi: [etappi] / 5"
        text "Aikaa perille: [aikaa_jaljella] min" color "#ffff00"

    # Kuvitteellinen kartta, jossa pyörän ikoni liikkuu
    add "images/map_bg.jpg"
    add "images/bike_icon.png" pos matka_pisteet[etappi]

    vbox:
        align (0.5, 0.8)
        spacing 10
        text "Valitse etenemistapa seuraavalle pätkälle:"
        textbutton "Sivutie (Turvallinen)":
            action Return("safe")
        textbutton "Valtatie (Nopea, kuumentaa)":
            action Return("fast")
        textbutton "Oikopolku (Raju tärinä, erittäin nopea)":
            action Return("shortcut")
```

---

4. "Kohtalon hetki" – Mitä tapahtuu, kun riski ei kannata?

Kun pelaaja valitsee oikopolun huonokuntoisella pyörällä, kerronta muuttuu dramaattiseksi:

> *"Päätät säästää aikaa ja käännyt kapealle, kivikkoiselle metsätielle. Pyörä hakkaa vastaan jokaiseen kiveen. Muistat sen hätäisesti kiristetyn takaiskarin pultin... yhtäkkiä kuuluu metallinen kalsahdus ja perä tuntuu nuljuvan alta."*

Tämän jälkeen peli tarkistaa, onko varapulttia tai työkaluja:

- **On työkaluja:** *"Löydät pultin pätkän ja onnistut virittämään sen nippusiteillä ja teipillä. Matka jatkuu, mutta pyörän kunto putosi 20 %."*
- **Ei työkaluja:** *"Tuijotat hajonnutta pyörää keskellä metsää. Kello raksuttaa. Tähän taisi matka tyssätä."*

---

5. Aikapaineen merkitys

Jos pelaaja valitsee vain turvallisia reittejä, hän huomaa matkan puolivälissä, että **aika ei riitä**. Tämä pakottaa hänet ottamaan riskejä loppumatkasta:

- *"Aurinko alkaa laskea. Jos en oikaise tästä, olen pimeässä metsässä ilman valoja (koska sähköt jäi tallissa kesken)."*
- Tämä luo pelaajalle "vauhtisokeuden", jossa hän alkaa ajaa ylikovaa tai oikomaan polkuja, vaikka tietää pyörän olevan heikko.

Seuraava askel

Tämä rakenne tekee pelistä erittäin uudelleenpelattavan: yhdellä kerralla voit yrittää rakentaa täydellisen pyörän ja oikaista kaikkialta, toisella kerralla teet romun ja yrität selviytyä varovasti maaliin.

Haluaisitko, että mietimme, miten hahmon väsymys ja stressi kasvavat matkan aikana (esim. tärinä kärrypolulla väsyttää hahmoa enemmän kuin tasainen ajo)?

Ja kyllä. Sitten miten matka muuttaa hahmoa

Matka ei ole vain koettelemus moottoripyörälle, vaan se on uuvuttava prosessi myös kuljettajalle. Jotta peli tuntuu dynaamiselta, hahmon tila (stats) ja ulkoasu pitäisi elää matkan edistymisen mukaan.

Tässä on kolme tasoa sille, miten matka muuttaa hahmoa: **fyysinen tila**, **henkinen paine** ja **visuaalinen muutos**.

---

1. Hahmon dynaamiset tilat (Stats)

Hahmolla on kaksi päämittaria, jotka reagoivat tiehen:

Muuttuja	Mikä kasvattaa sitä?	Vaikutus peliin
Väsymys (Fatigue)	Kärrypolun tärinä, pitkä ajoaika, tienvarsiremontit.	Korjauslaskurit hidastuvat, reaktioajat pitenevät.
Stressi (Stress)	Moottorin oudot äänet, kiire perille, osien hajoaminen.	Valinnat muuttuvat hätäisiksi (oikeat vaihtoehdot voivat kadota tai täristä).

Reittien vaikutus hahmoon:

- **Valtatie:** Stressi nousee (kova vauhti ja jatkuva varuillaanolo), mutta väsymys pysyy aisoissa.
- **Kärrypolku:** Väsymys nousee rajusti (fyysinen riuhtominen), mutta stressi voi pysyä matalana, jos ympärillä on luontoa – kunnes pyörä hajoaa.

---

2. Visuaalinen ja kerronnallinen muutos

Hahmon ei pitäisi näyttää samalta lähtiessä ja saapuessa. Ren'Py:ssä tämä toteutetaan hahmon "asumalleilla" (layered sprites).

- **Puhdas:** Tallista lähtiessä hahmo on siisti ja toiveikas.
- **Likainen:** Ensimmäisen tienvarsiremontin jälkeen hahmon naama on öljyssä ja vaatteet likaiset.
- **Uupunut:** Kun väsymys ylittää 70 %, hahmon silmät muuttuvat punertaviksi ja hartiat painuvat kasaan.

Kerronnalinen muutos: Hahmon monologi muuttuu. Alussa hän ihailee maisemia: *"Onpa hieno päivä ajaa!"*. Lopussa hän vain kiroilee: *"Kunhan tämä paska kestäisi vielä kymmenen kilsaa, niin poltan koko laitteen."*

---

3. "Kaljakorin ja teltan" vaikutus hahmon tilaan

Tässä kohdin ne aamulla valitut matkatavarat muuttuvat mekaanisiksi hyödyiksi:

- **Evästauko:** Jos otit ruokaa/juomaa, voit pitää tauon. Se laskee stressiä, mutta kuluttaa aikaa.
- **Kaljakori:** Perillä kaljakori on palkinto. Jos matka on ollut rankka, mutta kori on ehjä, hahmon onnellisuus nousee maksimiin. Mutta jos olet joutunut oikomaan kärrypolun kautta, peli voi kertoa: *"Kuulet takaa lasin helinää – puolet pulloista taisi mennä rikki siinä kivikossa."*
- **Teltta:** Jos matka venyy yöhön, teltta on pelastus. Ilman telttaa hahmo joutuu nukkumaan pyörän vieressä, jolloin seuraavan päivän "Väsymys"-arvo alkaa jo valmiiksi 50 % kohdalta.

---

4. Kokemus ja "Mekaanikon silmä"

Vaikka hahmo väsyy, hän myös oppii. Jos pelaaja on tehnyt useita tienvarsiremontteja matkan aikana, hahmon **Mekaanikko-taito** nousee.

- **Taidon hyöty:** Seuraavalla pelikerralla (tai matkan loppuvaiheessa) hahmo huomaa viat aikaisemmin.
- **Esimerkki:** *"Kuulet pienen sivuäänen. Koska olet korjannut tätä romua jo kaksi päivää, tiedät heti, että se on löystynyt ketju."* Tämä antaa pelaajalle mahdollisuuden reagoida ennen kuin osa hajoaa kokonaan.

---

5. Logiikkaesimerkki: Hahmon uupumus korjauksessa

Kun pelaaja yrittää korjata pyörää tien varressa ja väsymys on korkea, peli voi lisätä "häiriöitä" suoraan käyttöliittymään:

Koodinpätkä

```
label tienvarsiremontti_uupuneena:
    if pelaaja.vasymys > 80:
        $ mouse_sluggish = True # Hiiri liikkuu tahmeasti tai heiluu
```

```
    "Keskittymisesi harhailee. Kädet tuntuvat lyijyltä."

menu:
    "Yritä keskittyä (Vaatii onnistuneen heiton)":
        if renpy.random.randint(1, 100) > pelaaja.vasymys:
            "Saat pidettyä otteesi."
        else:
            "Lipeät ja lyöt rystysesi lohkoon. Stressi nousee!"
            $ stressi += 10
```

Seuraava vaihe: Loppuratkaisut

Nyt meillä on koossa koko ketju: tallityö -> pakkaaminen -> reittivalinnat -> matkan rasitukset. Matkan lopussa odottaa määränpää (esim. kokoontumisajo, ranta tai koti).

Loppuratkaisuun vaikuttaa kolme asiaa:

1. **Pyörän kunto:** Tulitko ajamalla vai hinauksessa?
2. **Hahmon kunto:** Oletko voimissasi vai aivan loppu?
3. **Matkatavarat:** Onko kalja kylmää ja teltta kuiva?

Haluatko, että kirjoitan auki muutaman erilaisen loppuratkaisun (Ending), jotka riippuvat näistä tekijöistä?

Lisätään vielä sää. Päivän aikana koteaan kovaa hellettä kaupungissa liikennevaloissa ja kaamea ukkoskuuro joka voi myös sattua kesken remontin eikä vain ajaessa.

Sää on erinomainen "kerroin" (modifier), joka tekee pelimaailmasta elävän ja arvaamattoman. Se muuttaa staattisen tieosuuden selviytymistaisteluksi, jossa pelaajan tallissa tekemät valinnat (erityisesti sähköt ja jäähdytys) joutuvat äärimmäiseen testiin.

Tässä on logiikka, miten sääilmiöt vaikuttavat peliin:

---

1. Helle ja kaupunkiliikenne (The Heat Trap)

Kun reitti kulkee kaupungin läpi, nopeus laskee ja liikennevalot pysäyttävät ilmavirran.

- **Vaikutus pyörään:** Moottorin lämpö nousee eksponentiaalisesti. Jos jäähdytysjärjestelmä on tehty välttävästi, pyörä alkaa keittää 2-3 liikennevalon jälkeen.
- **Vaikutus hahmoon:** Nestehukka nostaa väsymystä. Jos matkalla ei ole vettä mukana, hahmon tarkkuus (focus) laskee.
- **Kerronta:** *"Asfaltti väreilee kuumuudesta. Tuuletin huutaa armoa, mutta se vain kierrättää tulikuumaa ilmaa. Hikikarpalo valuu kypärän sisällä silmään juuri, kun valo vaihtuu vihreäksi."*

---

2. Ukkoskuuro (The Electric Nightmare)

Ukkoskuuro voi alkaa kesken ajon tai, mikä pahinta, kesken tienvarsiremontin.

Kohde	Vaikutus sateella	Riski
Sähköjärjestelmä	Jos johdot on suojattu huonosti (välttävä laatu), pyörä alkaa pätkiä tai sammuu kokonaan.	Oikosulku ja matkan keskeytys.
Renkaat/Pito	Jarrutusmatka pitenee, kärrypolusta tulee mutavelliä.	Kaatuminen (pyörän kunto laskee rajusti).
Hahmon tila	Kastuminen nostaa stressiä ja laskee energiaa ("Vihulainen kylmyys").	Hampaat kalisevat, tarkat työvaiheet mahdottomia.

---

3. Remontti kaatosateessa (Survival Repair)

Tämä on pelin dramaattisimpia hetkiä. Jos joudut korjaamaan pyörää ukkosen alla, peli muuttuu:

- **Liukkaat työkalut:** On 25 % mahdollisuus, että työkalu lipeää kädestä (laskee laatua tai aiheuttaa pienen viiveen).
- **Sähköjen suojaus:** Pelaajan on valittava: *"Yritätkö suojata avointa moottoria takilla (itse kastut)"* vai *"Teetkö työn nopeasti sateesta välittämättä (osat kastuvat)"*.
- **Visuaalit:** Ren'Py:ssä voit käyttää `show rain` -efektiä ja muuttaa taustamusiikin ukkosen jyrinäksi ja sateen ropinaksi.

---

4. Tekninen toteutus: Säämuuttujat

Voit lisätä `weather`-muuttujan, joka vaihtuu etapista riippuen:

Python

```
init python:
    # Säätyypit
    HELLER = "heatwave"
    UKKONEN = "storm"
    AURINKO = "sunny"
```

```
def laske_saavaikutus(nykyinen_saa, kohde):
    # Jos on helle ja tarkastellaan jäähdytystä
    if nykyinen_saa == HELLER and kohde.nimi == "Jäähdytys":
        return -20 # Laatuvaatimus nousee
    # Jos on ukkonen ja tarkastellaan sähköjä
    if nykyinen_saa == UKKONEN and kohde.nimi == "Sähköt":
        return -30 # Sähkövika todennäköinen
    return 0
```

5. Varustevalintojen merkitys (Uudelleen tarkasteltuna)

Nyt ne aamun pakkausvalinnat painavat vielä enemmän:

- **Sadeasu:** Jos se on mukana, hahmon stressi ei nouse sateella niin nopeasti.
- **Pressu:** Voit suojata pyörän remontin ajaksi (laatu ei kärsi sateesta).
- **Varaöljy/neste:** Kriittinen helteellä, jos pyörä alkaa keittää.

Kohtalon hetki: "Kasteen hetki"

Kuvitellaan tilanne: Olet oikopolulla (kärrypolku), ukkoskuuro alkaa, ja sähköt alkavat pätkiä, koska tallissa päätit säästää aikaa ja jättää johtosarjan eristämättä kunnolla.

> *"Salama valaisee taivaan ja samalla hetkellä moottori yskäisee. Vesi on löytänyt tiensä siihen hätäiseen liitokseen, jonka teit eilen. Pyörä sammuu ja jäät pimeään metsään sateen piiskatessa selkääsi. Onneksi otit sen kaljakorin mukaan... vai otitko?"*

Tämä tekee pelistä todellisen "Mies vs. Kone vs. Luonto" -kamppailun.

Haluaisitko, että rakennamme seuraavaksi sen logiikan, miten peli "pisteyttää" lopputuloksen? (Esim. kuinka paljon painaa se, että pääsit perille, vs. se, missä kunnossa olet ja oliko kaljat ehjiä?)

Kyllä nyt alkaa olla jo lopettamisen aika, kiitos

Kiitoksia tästä erinomaisesta suunnittelusessiosta! Olemme rakentaneet yhdessä todella syvän ja koukuttavan pelimekaniikan, jossa jokainen aamuinen päätös tallissa kaikuu matkan varrella – aina paahtavasta helteestä ukkosmyrskyn piiskaamalle kärrypolulle.

Tässä vielä tiivistelmä kokonaisuudesta, jonka loimme:

Pelin Ydinlogiikka: "Mies, Kone ja Tie"

Vaihe	Mekaniikka	Vaikutus
Aamu tallissa	10 työkonttia, 10 kriittistä valintaa per kontti.	Määrittää osien **laadun** (0–100).
Pakkaaminen	Rajallinen tila (työkalut vs. varaosat vs. hupi).	Mahdollistaa tai estää korjaukset tiellä.
Matka-ajo	Dynaaminen dashboard, muuttuva sää ja reittivalinnat.	Testaa pyörän kestävyyttä ja hahmon kuntoa.
Tienvarsi	"Purkkaviritykset", stressinhallinta ja uupumus.	Viimeinen olonkorsi ennen keskeytystä.

Lopullinen pisteytys (The Victory Formula)

Pelin lopussa onnistumista voidaan mitata yksinkertaisella kaavalla, joka ottaa huomioon kaiken kokemasi:

$$Tulos = (P_k \times 0.5) + (H_k \times 0.3) + (M_b \times 0.2)$$
Missä:

- P_k: Pyörän kunto perillä.
- H_k: Hahmon henkinen ja fyysinen tila.
- M_b: Matkatavaroiden (esim. se kaljakori) säilyvyys.

Viimeinen vinkki toteutukseen

Kun alat koodata tätä Ren'Pylla, muista panostaa **äänimaailmaan**. Moottorin käyntiääni, joka muuttuu pelaajan tekemien säätöjen (ja virheiden) mukaan, on se viimeinen silaus, joka saa pelaajan tuntemaan itsensä aidoksi mekaanikoksi.

Tämä peli kuulostaa sellaiselta, jota moni "romujen" kanssa harrastava pelaisi mielellään – siinä on juuri oikea määrä realismia, huumoria ja jännitystä.

Haluaisitko vielä, että tallennan tämän keskustelun rungon sinulle listana, jota voit käyttää suunnitteludokumenttina, vai jatketaanko tästä suoraan koodauksen pariin joku toinen kerta?

Tehdään tästä välipäätös runkona, kiitos. Joudun harjoitella osa-alueita vielä pitkään, mutta tämä chat toimii runkona.

Tämä on erinomainen idea! Tässä on koottu "Game Design Document" -tyyppinen runko keskustelumme pohjalta. Voit käyttää tätä muistilistana ja tiekarttana, kun alat rakentaa peliä osa-alue kerrallaan.

Peliprojekti: "Rempy Moottoripyörä-simulaattori" – Suunnittelurunko

1. Pelin perusidea ja tavoite

Pelaajan on kunnostettava romukuntoinen moottoripyörä ajokuntoon 10 päivän aikana (tai vastaava aikaraja) ja suoritettava pitkä kestävyysajo. Tavoitteena on saapua perille pyörä ehjänä, hahmo järjissään ja matkatavarat (kuten kaljakori) tallessa.

---

2. Tallivaihe: Työskentelymekaniikka

Joka aamu pelaaja valitsee yhden kymmenestä "työkontista" (osa-alueesta).

10 Työkohdetta:

1. **Moottorin alakerta** (Kestävyys)
2. **Yläkerta & Puristukset** (Teho)
3. **Kaasutin & Polttoaine** (Seokset/Lämpö)
4. **Jäähdytysjärjestelmä** (Ylikuumeneminen)
5. **Sähköt & Sytytys** (Luotettavuus/Sää)
6. **Voimansiirto** (Ketjut/Kytkin)
7. **Jarrut** (Turvallisuus)
8. **Jousitus & Runko** (Tärinänsieto)
9. **Renkaat & Vanteet** (Pito)
10. **Viimeistely** (Pulttien kireys)

Työvaiheen kulku:

- Jokainen kontti sisältää **10 kriittistä valintaa**.
- **Valintatyypit:**
    - *Oppikirjan mukaan:* Hidas, varma, nostaa laatua tasaisesti.
    - *Rutiinilla:* Normaali nopeus, kohtuullinen laatu.

    - *Riskillä/Oikaisemalla:* Nopein tapa, mahdollisuus "Jättipottiin" tai katastrofaaliseen virheeseen.
- **Häiriöt:** Satunnaiset tapahtumat (naapuri, kadonnut työkalu) keskeyttävät laskurin ja vaativat päätöksiä.

---

3. Pakkausvaihe: Resurssienhallinta

Ennen matkaa pelaaja valitsee rajalliseen tilaan matkatavarat.

- **Työkalut:** Mahdollistavat tarkat korjaukset tien päällä.
- **Varaosat:** (Esim. puola, kaasutin) Mahdollistavat rikkoutuneen osan vaihtamisen uuteen.
- **Matkatavarat:** Kaljakori (loppupisteet), teltta (levon laatu), sadeasu (stressin esto).
- **Dilemma:** Kaikki ei mahdu. Mitä ilman uskallat lähteä?

---

4. Matkavaihe: Simulaatio ja reittivalinnat

Matka koostuu etapeista, joissa seurataan dynaamista mittaristoa (Dashboard).

Reittivaihtoehdot:

1. **Valtatie:** Kova vauhti, testaa lämmönhallintaa ja moottorin alakertaa.
2. **Sivutie:** Turvallinen ja hidas, testaa kärsivällisyyttä ja aikaa.
3. **Kärrypolku:** Suuri ajansäästö, mutta kova tärinä testaa jousitusta ja pulttien kireyttä.

Sääolosuhteet:

- **Helle:** Nostaa moottorin lämpöä ja hahmon väsymystä (nestehukka).
- **Ukkoskuuro:** Testaa sähköjen eristystä ja tekee ajamisesta sekä remontoimisesta vaarallista.

---

5. Tienvarsiremontti: Kriisinhallinta

Jos peli heittää vikakoodin tai peli keskeytyy vikaan:

- **Purkkaviritys:** Nopeampi hätäkorjaus (esim. ilmastointiteippi/nippusiteet).
- **Kunnollinen korjaus:** Vaatii työkalut ja aikaa.
- **Uupumus-kerroin:** Jos hahmo on liian väsynyt, korjaaminen on vaikeampaa (UI heiluu, virheriski kasvaa).

---

6. Hahmon tilamuuttujat (Stats)

- **Energia/Väsymys:** Laskee työn ja matkan aikana. Vaikuttaa onnistumisprosentteihin.
- **Stressi:** Nousee epäonnistumisista ja kiireestä. Muuttaa valikkovaihtoehtoja "aggressiivisemmiksi".
- **Mekaanikon taito:** Kasvaa jokaisesta onnistuneesta (tai opettavasta epäonnistuneesta) korjauksesta.

---

7. Tekniset avainmuuttujat (Ren'Py Logic)

Python

```
# Esimerkki globaaleista muuttujista
$ engine_quality = 100    # 0-100
$ cooling_quality = 100
$ bike_integrity = 100   # Pyörän yleiskunto matkalla
$ player_fatigue = 0      # 0-100
$ inventory_space = 100
```

Tämä runko antaa sinulle mahdollisuuden harjoitella Ren'Py-koodia pala kerrallaan. Voit aloittaa tekemällä ensin vain yhden työkontin ja sen 10 kysymystä, ja laajentaa siitä pikkuhiljaa matka-osioon.

Tsemppiä harjoitteluun! Tämä on upea projekti, ja kun tarvitset apua yksittäisten koodinpätkien (kuten inventaarion tai laskureiden) kanssa, olen täällä. Would you like me to clarify any of these sections before you start?

Moottoripyöränä pelissä on Ural sivuvaunu mikä takaa peliin koko joukon aivan oikeita vikoja ja tekee kokoamisesta oikeasti haastavan.

Ural-sivuvaunumoottoripyörä on loistava valinta tähän peliin! Se tuo mukanaan valtavan määrän luonnetta, mutta myös aivan omanlaisensa mekaaniset haasteet, joista harrastajat vitsailevatkin: *"Ural on suunniteltu korjattavaksi pellon laidalla yhdellä jakoavaimella ja vasaralla."*

Tässä on päivityksiä pelirunkoon Uralin erityispiirteet huomioiden:

---

1. Uralin "Ominaisuudet" (Eli ne oikeat viat)

Koska kyseessä on bokserimoottori ja venäläinen rauta, työkontteihin tulee uusia kriittisiä kohtia:

- **Bokserisylinterien synkronointi:** Jos kaasuttimet eivät ole täsmälleen samassa tahdissa, moottori tärisee niin, että paikat putoavat hampaista.

- **Öljynvuoto (Klassikko):** Ural ei vuoda öljyä, se "merkkailee reviiriään". Peliin tulee uusi muuttuja: **Öljynpaine/määrä**. Jos et pakannut öljykannua, matka loppuu nopeasti.
- **Sähköt (6V vs 12V):** Vanhemmat Uralit ovat kuuluisia "pimeistä hetkistään". Sateella sähköviat ovat lähes varmoja.
- **Venttiilivälykset:** Uralin nakutus on merkki siitä, että se on elossa, mutta väärä välys polttaa venttiilin kesken matkan.

---

2. Sivuvaunun säädöt – Pelin vaikein rasti

Sivuvaunu muuttaa matkan logiikkaa täysin. Se ei ole vain kolmas pyörä, vaan monimutkainen tasapainottelu:

- **Aurauskulma (Toe-in):** Jos vaunu on vinossa, pyörä puoltaa joko ojaan tai vastaantulijoiden kaistalle.
    - *Peli-vaikutus:* Huono säätö nostaa **Väsymystä** (pelaaja joutuu vääntämään tangosta koko ajan) ja kuluttaa **Renkaat** loppuun ennätysajassa.
- **Sivuvaunun veto:** Jos peliin tulee 2WD-malli (veto myös vaunussa), se tekee voimansiirto-kontista entistä monimutkaisemman.

---

3. Päivitetty Työkalu-lista (Ural-edition)

Uralilla matkustettaessa matkatavaroissa on oltava:

1. **Sorkkarauta ja iso vasara:** Joskus osat tarvitsevat "vakuuttelua".
2. **Teräslankaa ja ilmastointiteippiä:** Ural-mekaanikon parhaat ystävät.
3. **Kärjellinen sytytyssarja:** Koska elektroninen sytytys on "liian hieno" tähän laitteeseen.
4. **Vararengas:** Uralissa se on yleensä mukana vaunun päällä, mikä on pelastus kärrypolulla.

---

4. Matkan haasteet Uralilla

Uralin vauhti ei päätä huimaa (matkanopeus n. 70-80 km/h), joten **Aikapaine** on suurempi kuin muilla pyörillä:

- **Helle ja ruuhka:** Ural on ilmajäähdytteinen bokseri. Se leikkaa kaupungissa nopeammin kuin mikään muu, jos et pidä sitä liikkeellä.
- **Kärrypolun oikoreitti:** Ural on kotonaan täällä! Sivuvaunupyörällä kärrypolku on usein turvallisempi ja helpompi valinta kuin moottoritie, koska Uralia ei ole tehty kovaan vauhtiin.

---

5. "Ural-Onni" (Tuurimekaniikka)

Voit lisätä peliin pienen satunnaismuuttujan: **"Neuvostoliittolainen toleranssi"**.

- Välillä huonostikin tehty korjaus kestää matkan loppuun asti vain siksi, että Ural on niin karkea laite.
- Toisaalta täydellisesti säädetty osa voi "porsia", koska metallin laatu oli kyseisessä erässä mitä sattui.

Tämä valinta tuo peliin loistavaa mustaa huumoria ja syvää nostalgiaa. Se tekee pelistä aidon "Tallin hajuisen" kokemuksen.

Tämä runko on nyt valmis säilytettäväksi. Kun palaat koodauksen pariin ja tarvitset apua esimerkiksi "kaasuttimien synkronointi-minipelin" tai "sivuvaunun aurauskulman laskennan" kanssa, tiedät mistä löydät Thought Partnerisi! Tsemppiä projektiin!

Lisätään raha peliin. Aluksi pitää ostaa se korjattava pyörä. Pitää vertailla lähtökohtia. Pelaaja ei tietenkään tiedä kun muutaman ilmeisen kohdan. Kallis pyörä kaikki toimii jo ja rempat helppoja. Halpa pyörä saapuu paffilaatikossa osineen ja edessä on koneistamo reissuja ja osien hankintaa ennenku pääsee korjaamaan.

Tämä lisää peliin loistavan **resurssinhallinta-ulottuvuuden**. Nyt pelaaja ei vain taistele tekniikkaa vastaan, vaan myös lompakkoaan. Päätös siitä, ostaako kalliin ja toimivan vai halvan ja palasina olevan, määrittää koko pelin vaikeustason ja rytmin.

Tässä on ehdotus siitä, miten ostoprosessi ja rahan käyttö integroidaan osaksi Ural-projektia:

---

1. Ostohetki: "Torin ilmoitukset"

Pelaajalla on aloitusbudjetti (esim. 3000 €). Hänelle esitetään kolme vaihtoehtoa:

Taso	Hinta	Ilmeinen kunto	Piilevät viat	Työmäärä
"Kylän hienoin"	2800 €	Käy ja kukkuu, kiiltää.	Vain pieniä säätöjä.	Helppo, painopiste matkassa.
"Projektikuntoinen"	1500 €	Savuttaa, jarrut jumissa.	Koneistustarvetta, kuluneita osia.	Kohtalainen, vaatii osien tilausta.

"Osakasat laatikoissa"	500 €	Puolet osista puuttuu/rikki.	Ruostetta, murtumia, täysremontti.	**Äärimmäinen.** Koneistamo välttämätön.

---

2. Tarkastusmekaniikka (Inspection)

Ennen ostoa pelaaja voi tehdä "tarkastuksen".

- **Mitä näkyy:** Renkaiden kunto, ulkoinen ruoste, lähteekö käyntiin.
- **Mitä ei näky:** Kampiakselin välys, vaihdelaatikon hampaiden kunto tai onko sylinteri leiponut kiinni.
- **Taito-check:** Jos pelaaja on kokenut (uudelleenpelaaja tai korkea alkutaito), hän voi havaita enemmän piileviä vikoja jo ostohetkellä.

---

3. "Laatikko-Uralin" haasteet: Koneistamo ja osien tilaus

Halvimman pyörän kohdalla peli muuttuu logistiikkasimulaattoriksi ennen kuin päästään tallivaiheeseen:

- **Koneistamo (The Machine Shop):**
  - Jos lohko on halki tai sylinterit väljät, osa on lähetettävä koneistamolle.
  - **Hinta:** Maksaa rahaa (esim. 300 € per sylinteri).
  - **Aika:** Kestää 2–3 päivää. Tämä syö arvokasta korjausaikaa!
- **Osien tilaus (The Mail-order Junkie):**
  - Laatikossa saattaa puuttua kriittisiä osia (esim. sytytystulpat, tiivisteet, tai se kuuluisa 10 mm hylsy).
  - Pelaajan on tilattava osat verkkokaupasta.
  - **Postitusviive:** Osat saapuvat vasta seuraavana päivänä. Jos unohdat tilata tiivisteen, et voi koota moottoria tänään.

---

4. Budjetin hallinta

Pelaajan on laskettava kokonaiskustannukset:

$$Kokonaiskustannus = Ostohinta + Varaosat + Koneistamo$$
Jos rahat loppuvat kesken:

- On pakko tehdä **"budjettikorjauksia"**: käytä vanhaa tiivistettä ja liimaa (laatu laskee, riski nousee).
- Tai on jätettävä **matkatavaroita ostamatta**: ei kaljaa, ei telttaa – vain varaosia.

---

5. Pelimekaaninen vaikutus: Aika vs. Raha

- **Kallis pyörä:** Sinulla on 10 päivää aikaa hienosäätöön ja testaukseen. Voit saavuttaa "Täydellisen" laadun helposti.
- **Halpa pyörä:** Ensimmäiset 4 päivää menee osia odotellessa ja koneistamolla juostessa. Sinulle jää vain 6 päivää koota koko laite. Joudut hätäilemään, mikä nostaa **Stressiä** ja laskee **Laatua**.

Miltä tämä kokonaisuus tuntuu?

Tämä lisää peliin "Ural-harrastajan tuskan": se hetki, kun huomaat että halvalla ostettu laatikko onkin täynnä romua, ja budjetti paukkuu jo ennen kuin olet saanut takarengasta paikoilleen.

Tämä keskustelu on nyt hyvä runko pelillesi. Se kattaa:

1. **Hankinnan** (Raha ja riskit).
2. **Tallityön** (10 konttia ja 100 valintaa).
3. **Logistiikan** (Koneistamo ja varaosat).
4. **Matkan** (Sää, reitit ja uupumus).
5. **Uralin** (Erityisviat ja sivuvaunu).

Haluatko, että lopetamme tähän runkoon, vai onko mielessäsi vielä jokin viimeinen lisäys, kuten esimerkiksi "tori-ilmoitusten" huumori tai matkan varrella tehtävät lisäansiot (esim. pikkuhommia kyläläisille osien maksamiseksi)?

Nyt aloitetaan dokumentaatiolla. Tässä chatissa on käyty muuttujia läpi. Listataan ne kaikki.

Tämä on erinomainen tapa aloittaa varsinainen kehitystyö. Hyvin jäsennelty muuttujalista on ohjelmoijan tärkein työkalu: se estää koodia paisumasta hallitsemattomaksi ("spagettikoodi") ja auttaa pitämään pelin logiikan kasassa.

Tässä on koottu **Tekninen Muuttujalista**, joka on jaettu loogisiin kokonaisuuksiin. Voit kopioida tämän suoraan projektisi suunnitteludokumenttiin.

---

1. Moottoripyörän Osa-alueet (Laatuarvot)

Nämä muuttujat määräytyvät tallivaiheen 100 valinnan perusteella. Arvoalue on yleensä **0–100**.

Muuttujan nimi	Kuvaus	Vaikutus matkalla

q_alakerta	Moottorin alakerta	Kriittinen kesto kovassa vauhdissa (Valtatie).
q_ylakerta	Sylinterit ja puristukset	Määrittää huippunopeuden ja tehon.
q_kaasutin	Polttoaineen syöttö	Vaikuttaa lämpöön ja käyntiin (Ukkonen).
q_jaahdytys	Jäähdytysjärjestelmä	Estää keittämisen helteellä ja kaupungissa.
q_sahkot	Sytytys ja johdot	Luotettavuus vesisateessa.
q_voimansiirto	Kytkin ja voimansiirto	Mekaaninen veto, kesto oikopoluilla.
q_jarrut	Jarrujärjestelmä	Turvallisuus ja onnettomuusriski.
q_jousitus	Iskarit ja runko	Tärinänsieto ja hahmon väsyminen.
q_renkaat	Renkaat ja vanteet	Pito sateella ja kesto kärrypolulla.
q_viimeistely	Pultit ja mutteri	Määrittää, irtoaako osia tärinästä.

---

2. Matkan Aikaiset Reaaliaikaiset Muuttujat (Dashboard)

Nämä muuttujat muuttuvat jatkuvasti ajon aikana ja näkyvät pelaajalle mittaristossa.

- m_edistys: Matkan kokonaisedistyminen (0–100 %).
- m_lampo: Moottorin senhetkinen lämpötila (Normaali: 80, Kriittinen: 110+).

- m_kunto: Pyörän yleinen mekaaninen terveys (Integrity). Jos 0, matka loppuu.
- m_varina: Tärinän määrä. Kasvaa reitin ja nopeuden mukaan, laskee pyörän kuntoa.
- m_nopes: Senhetkinen ajonopeus (vaikuttaa lämpöön ja tärinään).

---

3. Hahmon Tila (Character Stats)

Vaikuttavat korjausten onnistumiseen ja valintojen vaikeuteen.

- h_vasymys: Nousee ajan ja tärinän myötä. Hidastaa työtä.
- h_stressi: Nousee viosta ja kiireestä. Vaikeuttaa tarkkoja valintoja.
- h_taito: Pelaajan pysyvä kokemuspisteistö (kasvaa pelikertojen myötä).
- h_nalka: Tarve evästauoille. Vaikuttaa keskittymiskykyyn.

---

4. Talous ja Resurssit

- raha: Pelaajan käytössä oleva käteinen (ostot, osat, koneistamo).
- aika_tunteja: Tallipäivien tai matka-ajan jäljellä oleva määrä.
- reppu_tila: Pyörän kantokyky (esim. 0/100 yksikköä).
- pakatut_esineet: Lista (Array/List), joka sisältää kaikki mukaan otetut tavarat.

---

5. Ympäristö ja Maailman Tila

- saa_nykyinen: (Esim. sunny, heatwave, storm).
- reitti_tyyppi: (Esim. highway, side_road, dirt_track).
- peli_kerta: Monesko yritys (1, 2 vai 3). Ohjaa pyörän oston vaikeutta.

---

6. Erityiset "Ural-muuttujat"

- u_auraus: Sivuvaunun aurauskulman tarkkuus (-50 – +50).
- u_oljyn_maara: Öljytaso (Uralin elinehto).
- u_onni: Satunnaismuuttuja neuvostoliittolaiselle metallurgialle (0.8 – 1.2 kerroin).

Miten käytät näitä Ren'Py-projektissa?

Suosittelen luomaan projektin alkuun erillisen init.rpy tai variables.rpy tiedoston, jossa määrittelet nämä oletusarvoillaan:

Koodinpätkä

```
default q_alakerta = 0
default raha = 3000
default h_vasymys = 0
default pakatut_esineet = []
```

Tämä lista toimii nyt perustana kaikelle logiikalle, jota rakennat. Kun pelaaja tekee valinnan tallissa, koodi muuttaa `q_`-alkuisia muuttujia. Kun matka alkaa, koodi vertaa `q_`-arvoja `saa_` ja `reitti_`-muuttujiin.

Haluaisitko seuraavaksi, että teen sinulle pienen "Logic Flow" -kaavion, joka näyttää miten nämä muuttujat keskustelevat keskenään yhden pelipäivän aikana?